(2)

Report No. 7172

Radio-Parameter Selection Algorithm for Receiver-Directed Packet-Radio Networks (SRNTN-73)

1990

Julio Escobar

Prepared By:

BBN Systems and Technologies Corporation
10 Moulton Street
Cambridge, MA 02138

Prepared for:

DARPA/ISTO
1400 Wilson Bl.
Arlington, VA. 22209

Sponsored by:

The Defense Advanced Research Projects Agency
Under Contract No. MDA-903-83-C-0173

Contents

List of Figures

List of Tables

Abstract

This report describes the Parameter Selection Algorithm for the SURAP 4 packet radio network and the basis for its design. It also presents a Semi-Markov model of the algorithm operation. The algorithm adaptively adjusts three radio parameters: transmitter power, FEC coding rate, and channel bit rate. These parameters can increase the perceived signal-to-noise ratio on the channel and are treated as gain mechanisms. The algorithm must maintain robust radio links between nodes whenever possible, while minimizing interference with other network transmissions. We present one heuristic approach to balancing these two goals. This approach uses barely sufficient gain to maintain the packet error probability below an acceptable threshold value. A threshold probability of 0.1 was chosen. Bit error statistics provided by the sequential decoder and error feedback packets assist the process of determining the appropriate change of gain as noise levels vary. The algorithm can adapt at a fast rate, sometimes on the order of one packet time. It maintains acceptable packet error probability and can recover from estimation and decision errors.

We present a Semi-Markov model for the analysis of the algorithm. The model can be used to compute the steady state packet error probability and packet transmission overhead due to noise. The model is general enough to include a stationary model of transceiver blocking probability, dynamic CDMA interference, and alternative algorithms based on similar operating principles. A maximum of 38 Markov states are required to model precisely the algorithm's steady-state behaviour. The structure of the state space and the transition probability expressions is straightforward. The transition probability matrix is sparse.

Although the model does not incorporate bit error statistics, we have proposed a way to model the effect of these statistics without compromising the simplicity of the Semi-Markov description.

1. Introduction

Objectives An adaptive physical layer algorithm in a Packet Radio Network is essential for the network survivability. The quality of the radio channels is subject to rapid changes resulting from mobile nodes and jamming. If radio hardware is designed for a worst case scenario, the throughput for normal situations will be greatly impaired. If the hardware is designed for an average situation, the network risks partition under stress because of poor radio links. The Parameter Selection Algorithm provides adaptive control of physical layer radio parameters.

This report describes the Parameter Selection Algorithm for the Survivable Adaptive Networking (SURAN) packet radio project sponsored by the Defense Advanced Research Projects Agency (DARPA). The Parameter Selection Algorithm is an adaptive algorithm designed to maintain radio links between nodes in the network under conditions that include changing noise, interference, and jamming. The Low-cost Packet Radio (LPR) hardware employed in the SURAN project provides a choice of transmitting power levels, convolutional coding rates, and channel transmission rates. The Parameter Selection algorithm adjusts these parameters adaptively to maintain connectivity among radios and minimize interference with other transmissions. The algorithm plays an important role in establishing and terminating links between nodes.

The algorithm is designed to operate in the SURAN protocol (SURAP) suite number four (SURAP 4). SURAP 4 employs receiver-directed spread-spectrum code-division multiple-access (CDMA) transmissions for increased throughput. The suite uses a distributed congestion control algorithm that is capable of achieving Max-Min Fair flow control over a large class of situations. It also employs a security architecture to prevent or detect misbehaving nodes. SURAP 4 is an evolution step from earlier SURAP versions developed under the same DARPA contract. An overview of the SURAP 4 architecture can be found in [4].

The Parameter Selection Algorithm can be modified for other packet radio architectures. For example, Rockwell International has developed a version of the Parameter Selection Algorithm taylored to the the SURAP 3 architecture. SURAP 3 does not use code-division multiple-access.

This report is divided into six sections. Section 1 is the introduction. It outlines the design objectives and the approach followed. It also contains a brief survey of work relevant to the analysis and design of the Parameter Selection Algorithm. Section 2 contains a description of SURAP 4 algorithms relevant to the design of the Parameter Selection Algorithm. Section 3 describes the nature of the radio channel and its implications on the architecture of the algorithm. Section 4 outlines the design principles and describes the algorithm in detail. Section 5 presents analysis and simulation results of the performance of the algorithm. This section includes a Semi-Markov model of the algorithm operation. Section 6 contains conclusions and recommendations for future work.

Design Approach The problem of selecting the right set of radio parameters for an entire packet radio network is complex. Because airwaves are a broadcast medium, the transmission parameters for one radio channel affect the transmission and reception of all other radios within range. The choice of transmission parameters also affects the throughput of the transmitting radio. Lower Forward Error Correction (FEC) rates and lower channel bit rates decrease bit error rates but imply longer transmission times, hence limiting the throughput of the transceiver. Another problem is the interactions between the Parameter Selection Algorithm and the other network algorithms. Behavioral changes in the physical layer test the tolerance of higher-layer algorithms and protocols. Besides addressing these problems, the algorithm must be distributed so that it can provide quick and reliable responses. It must also be robust so that the network can survive under adverse conditions.

To simplify the design problem we optimized the radio channel parameters locally, instead of globally. The main goal of the algorithm is to maintain a robust radio link by adapting the links's parameters to the channel noise level. The algorihtm controls the link parameters so that the physical layer of the netwok appears stable to the algorithms in higher layers of the network. It is of particular importance to maintain a low average number of transmission attempts per packet forwarded on the links. Thus the main criterion for adapting channel gain is the estimated packet error rate that results from noise or interference. Parameter selections are characterized in terms of the relative gain over the channel signal-to-noise ratio that they afford. Gain increases are the result of increases in transmitter power and decreases in FEC rate and channel bit rate. A gain that exceeds the value required to maintain a robust link unnecessarily penalizes the throughput of all radios in the vicinity. Thus another important goal of the algorithm is to minimize the gain required to maintain a robust link.

The algorithm takes advantage of an estimate of channel bit errors provided by the receiver hardware when FEC is in use. The bit error information can sometimes provides an estimate of the state of the channel within the order of a packet time. The LPR hardware also provides a measurement of noise energy and signal plus noise energy when a packet is being received. The Parameter Selection Algorithm has not incorporated this information into its decision mechanism.

Algorithm Overview In the SURAP network, three radio parameters —transmitter power, FEC coding rate (convolutional codes), and channel bit rate— can be independently adapted in discrete steps and between packets. In the network design we recommend using FEC in all transmissions since the coding overhead incurred for the weakest coding rate (8/7) is minimal and we obtain bit error statistics form the sequential decoder.

A *parameter triplet* is a specific combination of the three channel parameters available for algorithm control. The Parameter Selection Algorithm selects the operating parameter triplet based on channel feedback after each transmission. Triplets are ranked in order of Signal-to-Noise Ratio (SNR) gain. A triplet is also known as the *gain state*. The greater the SNR gain of this state the greater is its adverse impact on neighbors. As the algorithm provides a satisfactory link it also tries to minimize the link gain. An SNR gain is satisfactory if it is sufficient for a maximum-length packet to maintain a packet error probability below a threshold parameter value (set to 0.1 for SURAP 4) when the packet is transmitted over the link in the absence of contention. This probability aims at maintaining a fairly predictable behavior of the link[1] and providing a rational interaction with higher-level protocols.

[1]Retransmissions are considered the main reason for link delay

SNR gain on a link is decreased by a transmitting node after enough successful transmissions have been observed. SNR gain is increased after failed transmissions or error acknowledgments[2]. Bit error counts, when available, are included in packet acknowledgments for the sending node. The bit error count is used to estimate the change in SNR gain required for satisfactory link operation. Bit error counts are also used to estimate the packet error probability resulting from noise. This estimate is factored out from attempts-per-packet computations in the Congestion Control Algorithm.

Literature Survey Spread spectrum is used in the SURAN packet radio network to protect the network from jamming, but it can also provide Code Division Multiple Access (CDMA) capability. Overlapping transmissions need not result in a collision, in the Aloha sense, since receivers synchronized with one code sequence can reject the energy from an interfering code sequence (code capture). Several authors have characterized the CDMA radio environment. [11] gives asymptotically tight upper and lower bounds for the probability of error in a CDMA environment, as a function of the number of overlapping transmissions, the channel signal-to-noise ratio, and the spreading factor for the code. We have used these results to discuss CDMA interference. In [7] these bounds are improved by deriving the probability density function for the interference signal. The effect of transmitter power is also incorporated. Similar work has been presented in [6] and [10], which gives simpler computation procedures for evaluating arbitrarily tight bounds. [10] includes jamming signals. A general treatment of CDMA performance can be found in [9]. In Section 3.1 we use the treatment of the "near-far" problem from [9] when discussing the expression for error probability in a CDMA environment and the effect of jamming.

The channel model in [16] incorporates CDMA interference and transceiver contention into the probability of failed transmission. The model specifically applies to a fully connected packet radio environment that uses Direct Sequence Spread Spectrum Multiple Access and Viterbi decoding for convolutionally encoded packets. Specific work treating error probabilities for a CDMA packet radio with convolutional codes and Viterbi decoding also appears in [12]. This works derives a bound on packet error probability that is an increasing function of packet length. System throughput is investigated under specific assumptions. [14] treats the throughput of ARQ strategies in radio links using sequential decoding. The optimal value for the time-out parameter of the sequential decoder is derived and the corresponding throughput is presented. Throughput of multihop packet radio models has been analyzed and evaluated in [13], [1]. Routing strategies based on least-interference computations are investigated in [15]. Transmission power can be adjusted and a distributed algorithm is used to compute routes and link powers which minimize interference among adjacent links. This work is also an excellent source of references on work in dynamic power adjustment for the multihop packet radio environment. A similar approach to the routing strategies in [15] is followed by [2] for computing optimal paths over a multihop CDMA packet radio network.

[2] By looking into the sender ID field of a packet with errors we can often determine which neighbour has sent the packet. By virtue of its small length, this field is less prone to errors than the packet as a whole. The receiving node can then return a special error acknowledgment packet.

2. Surap 4 Algorithms

2.1 Architecture

A general understanding of several SURAP 4 algorithms will be useful in the discussion of the design of the Parameter Selection Algorithm. In this section we give a simplified overview of the relevant algorithms. These protocols are documented in detail in [8, 17, 4, 5].

The SURAP 4 architecture uses Receiver-Directed Direct-Sequence spread-spectrum transmission. This means that a packet is encoded with a pseudorandom sequence recognizable only by its intended receiver radio. The throughput improvements obtained with this CDMA scheme are documented in [18]. There is also a *broadcast* sequence which is recognizable by all radios, providing for multicast capability. At the link layer we can group the protocols, or algorithms, into Channel Access Protocols, Link Flow Control Protocols, and Physical Layer Protocols. At the network layer we are only interested in the Congestion Control algorithm and the Routing algorithm.

During the course of writing this report, some of these algorithms are being modified as greater simulation experience is gained. Thus the exact design of some of these algorithms may be different than the one described here. So far none of these modifications seems major enough to alter the conclusions of this report. Since the modifications themselves are under experimentation we have chosen not to incorporate them in the report. This section remains a very accurate description of the algorithms.

Channel Access Channel Access in the SURAP 4 broadcast medium is regulated by the following protocols and algorithms:

- Spacing Algorithm
- Moment of Silence Algorithm
- Retransmission Protocol
- Alternate Routing Protocol

The *Spacing Algorithm* determines a minimum wait between packets transmitted on a link. The intention is to avoid sending a new packet before the neighbor is ready for it. The wait is typically short and of low variance. The *Moment of Silence Algorithm* gives acks high priority in the network so that network resources can be freed quickly and re-used. The *Retransmission Protocol* regulates retransmission waits for unacked data packets. It also limits to six the maximum number of packet transmission attempts before discarding the packet. Retransmission waits are typically larger than, but of the same order as, the Spacing waits. The *Alternate Routing Protocol* specifies that after four failed transmission attempts, a packet is broadcast using the broadcast spread-spectrum sequence. Thus any radio within range can help in forwarding the packet to the next hop in its path to the final destination.

Link Flow Control Link Flow Control is implemented to confine the congestion effects of high-rate traffic. The following protocols enforce Link Flow Control:

- Link Window Protocol
- K-Buffering Protocol

The *Link Window Protocol* prevents neighbors from being inundated with packets. This protocol requries neighbors to wait for an acknowledgement before forwarding a new packet. The *K-Buffering Protocol* limits to K the number of packets that a neighbor can buffer from a given node, preventing single neighbors from monopolizing all buffers. It also imposes increasingly large retransmission waits for neighbors at their K limit.

Link protocols normally shield a network region from high-rate traffic, making it difficult for the Congestion Control Algorithm to detect the source of the flow bottlenecks. It is imperative that we prevent link protocols from taking over the functions of the Congestion Control Algorithm because this algorithm is better suited to controlling the flow of traffic. The waits imposed by the Spacing Algorithm, the Retransmission Protocol, and especially the K-Buffering Protocol had to be redesigned so that natural packet flow was seriously impeded only in extreme situations.

Physical Layer Protocols At the Physical Layer we are only concerned with two protocols:

- Parameter Selection Algorithm
- Link Up/Down Protocol

We are not concerned with protocols determining packet format, modulation and other physical layer functions. The subject of this report is the *Parameter Selection Algorithm*. The *Link Up/Down Protocol* is closely related to the Parameter Selection Algorithm. It uses information generated by the Parameter Selection Algorithm to tell the network when to establish and terminate radio links.

Network Layer Algorithms At the Network Layer, we are interested in two algorithms:

- Congestion Control Algorithm
- Routing Algorithm

The *Congestion Control Algorithm* regulates end-to-end traffic flow in a distributed manner, by monitoring network resources at each node, independently, and adjusting source flows iteratively until all resources are utilized at or below a target value. Typical resources are transceiver utilization and buffer utilization. All flows sharing a resource are throttled equally, and the algorithm is capable of providing Max-Min Fair flow control over a large class of situations. Exceptions are detailed in [5]. The *Multi-Class Routing Algorithm*, is a distributed minimum-hop routing algorithm. The algorithm is designed to allow packet forwarding through links that have poor performance, but are the only alternative to a destination. The algorithm chooses the shortest-hop route consisting entirely of good-quality links. If one does not exist, then it chooses the shortest-hop route consisting of links of any quality. The Link Up/Down Protocol can be viewed as an interface between the Parameter Selection Algorithm and the Multi-Class Routing Algorithm.

2.2 Channel Access Protocols

In the Receiver Directed architecture proposed, all acknowledgments are active. A special packet is created by the receiving node and broadcasted back to the sending node. The *Moment Of Silence Algorithm* seeks to give high priority to these ack packets. It improves throughput by allowing nodes to free resources quickly. Since active acknowledgments are of short length, the channel time reserved for them represents minimal overhead. After forwarding a data packet, the sender node refrains from transmitting for an interval of τ_{MOS} time units. This interval is known as the moment of silence. τ_{MOS} is a network parameter equal to the time required for the receiver node to prepare and start sending the ack. The moment of silence at the sender can only be preempted by some packet reception. A receiving node gives highest priority to preparing and transmitting the ack upon receiving a data packet, so as to comply with the moment of silence at the sender. Data transmissions or retransmissions are delayed until the ack is sent.

The *Spacing Algorithm* enforces a minimum wait between successive first transmissions of a data packet. The spacing wait allows for packet processing, packet transmission plus moment of silence, and ack reception at the receiver node. The wait is enforced from the end of an ack reception or the end of the moment of silence at the sender node, whichever comes first. Computation of the wait, or spacing value, is done at the receiver node and is communicated back to the sender via the ack[1]. In parallel with this algorithm, the *Retransmission Protocol* sets a retransmission timer as soon as a packet transmission ends. If the ack for a transmission arrives before this timer expires, the residual spacing wait is observed before transmission of the next data packet; otherwise a second transmission is attempted and a new retransmission timer is set. If six transmissions fail, the buffer containing the packet becomes available for any other packets requiring it. The timer w_{n+1} for scheduling transmission $n+1$ is computed according to

$$w_{n+1} = k\,\tau_{MOS} + n(\text{MAX\_PKT\_TIME} + \tau_{MOS})\,\text{RANDOM} \qquad (2.1)$$

k is a small integer constant, chosen equal to 3 in our design. MAX_PKT_TIME is the longest duration of a packet on the channel (approximately 60 milliseconds with channel rate 100 Kbps and with 1/2 fec rate). n is the index for the most recent transmission. RANDOM is a random variable uniformly distributed between 0 and 1. The formula above is similar to linear back-off retransmission strategies, where the retransmission wait increases with the number of retransmissions. Using MAX_PKT_TIME is a heuristic way of accounting for large packets on the channel. Its randomness prevents synchronization among retransmitting nodes. A special retransmission wait is aplied when the transmission failure is due to bit errors. The receiving node acknowledges the packet through an *error acknowledgment*. The transmitting node uses a wait equal to the wait in (2.1), excluding the deterministic factor. The *Alternate Routing Protocol* enhances the survivability of the network by providing the capability to route around failed or ill-functioning nodes and links without invoking the Routing Algorithm. After three failed transmission attempts, a node uses the broadcast spread-spectrum sequence to send the packet. Radios within range, which receive this packet, are allowed to acknowledge it and forward it to its final destination.

[1] If the receiver node is the final destination for a packet, the spacing value computed is smaller because there is no need to relay the data packet

2.3 Link Flow Control

The *Link Window Protocol* simply specifies a link window of one. Acks must be received for each packet before the next one can be forwarded. Link windows greater than one are not implemented because of the limited resources in the nodes and the relative reliability of ack reception in the architecture, which results from the Moment of Silence Algorithm. The subject of optimal link window size in SURAP 4 has not been studied in detail.

K-Buffering regulates the use of buffers in a node. A neighbor relaying traffic through a node is allowed to use a maximum of K packet buffers in the node at any given time. The host attached to a node is allowed to use a maximum of two buffers. If new packets from a neighbor at its K limit are received, they are not buffered. A special ack packet is returned to the neighbor, requiring it to observe an exponentially increasing waiting time for successive retransmissions, until one or more of its packets is forwarded. Periodically, a time-window averaged service time for packets from each neighbor is measured. The waiting time for K-Buffering is based on this average. The total wait, however, is never increased past a maximum value (1.5 seconds in our design). This flow control mechanism is far more drastic than Spacing or Retransmission waits and is only invoked when the K limit is reached.

2.4 Physical Layer

The *Parameter Selection Algorithm* described in Section 1, adaptively controls radio channel parameters to provide a robust communication link in spite of changes in the channel characteristics.

The *Link Up/Down Protocol* specifies rules for establishing and terminating a radio link. In this report the Link Up/Down Protocol is treated as if it were part of the Parameter Selection Algorithm, and is described along with the algorithm.

2.5 Network Layer

The *Congestion Control Algorithm* regulates end-to-end flows in the network, preventing or eliminating congestion and attempting to provide Max-Min Fair flow allocation among sources.

A *path ration* is the maximum rate allowed by a node to any flow whose path passes through this node. Each node computes a path ration by monitoring transceiver utilization, CPU utilization, buffer use, and potential output-link throughput. The resource utilizations are compared to thresholds to determine whether the path ration should be increased or decreased. One threshold value is very relevant to this report. The threshold value on the average number of transmission attempts per packet is set at 1.5 attempts. This threshold limits transceiver contention. The minimum path ration among nodes in a path is communicated back to the flow source, using a distributed algorithm that piggy-backs rations in link-level acknowledgments. The flow rate on each route is then throttled according to the received minimum path ration. In this way all sources sharing a bottleneck resource are throttled equally. Sources not affected by a bottleneck are not throttled.

The *Routing Algorithm* determines end-to-end routes in the network according to two criteria. The first is the quality of links. The second is the hop count on the route; the minimum-hop path is chosen. A link can be either *good* or *bad*. The link type is determined by the success of the Parameter Selection

Algorithm to establish communication between nodes. Acceptable links are deemed good. Unacceptable links, where communication is still possible but only after many retransmissions, are deemed bad links. When communication seems impossible the link is terminated if in existence, or rejected if requesting existence. A route with one or more bad links is a second class route. A route with no bad links is a first class route. A node first attempts to route packets by using minimum-hop first class routing. If no such route is available, it uses the minimum-hop second class route.

3. Channel Considerations

In Section 3 we discuss the nature of the radio channel. Subsection 3.1 discusses channel noise and jamming, and the probability of packet error as a function of these factors. Subsection 3.2 discusses channel contention. Subsection 3.2 also discusses the effect of channel access on the probability of packet reception failure. The effect of interference from packets transmitted by nearby nodes is also included in this subsection since it is closely linked to the contention process.

3.1 Noise and Jamming

Radio channel Reference [3] describes the LPR hardware. We will work with only the subset of the hardware capabilities that is relevant to the Parameter Selection Algorithm. The algorithm controls three channel parameters: average transmitter signal power, channel bit rate, and FEC coding rate. The possible values of these parameters appear in Table 3.1. The hardware provides a choice of four power levels. It also provides a choice of two channel bit rates. The product of these two parameters determines the *symbol* energy. By symbol we mean the unit of reception in the receiver hardware. For uncoded packets, each data bit constitutes a symbol. For coded packets, a data bit produces one or more *code bits* and each code bit constitutes a symbol. The parameter "channel bit rate" refers to the symbol rate. An equivalent term is "baud rate." The hardware provides a choice of no FEC or three convolutional coding rates. A sequential decoder is used for decoding. The code rate is defined as the ratio of data bits to code bits. In general, the lower the rate is the stronger the error correction capability of the code is. The rate at which bits are transmitted in the channel is determined only by the channel bit rate and is independent of the coding chosen.

In SURAP 4, as in previous versions of SURAP, each symbol (data bit or code bit) is transmitted via a spread-spectrum signal. The symbol modulates a sequence of *chips*. Chips are polarity bits. The chip rate is higher than the symbol rate. The chip rate is fixed at 12.8 Mega-chips per second. The ratio of chip rate to symbol rate is known as the *spreading factor* N. For each symbol, the transmitter sends a sequence of N chips. For the 100 Kbps channel bit rate, N = 128. For the 400 Kbps channel bit rate, N = 32. The N-chip sequence is the spread-spectrum sequence or spread-spectrum code. It is a pseudo-random sequence designed to provide anti-jam and multiple access capabilities. Specific use of this multiple access capability is one of the improvements introduced in SURAP 4.

Minimum Shift Keying (MSK) modulation is used to transmit the chips. MSK can be regarded as a bandwidth efficient technique for Phase Shift Keying (PSK) modulation. M-ary PSK assigns one carrier signal phase, out of M possible phases, to each of the M possible signal values being transmitted. Since our chips are polarity bits, the MSK technique is used on *binary* PSK. Because of the polarity nature of the signals employed, the MSK modulation that is used falls into the category of *binary antipodal* signal sets, whose probability of error characteristics are well understood. For reception of a symbol, the

Table 3.1: Radio Parameter Values

Power (mW)	FEC rate	Channel rate (Kbps)
20	7/8	400
125	3/4	100
800	1/2	
5×10^3		

receiver circuit synchronizes with the carrier and makes a correlation operation, across the N chips of a received symbol, with a local replica of the pseudo-random sequence recognized by the receiver. If there is a match, the polarity of the correlation output determines which of 1 or 0 is the channel bit received. If there is no match, the output will typically behave like noise and will be rejected by the circuitry. In FEC operation, the received symbol sequence is handed over to the sequential decoder to recover the data bits that make up the received packet.

A problem affecting the channel bit error probability is multipath interference. Signal reflection from objects in the geographical area can cause the existence of multiple propagation paths between a sending transceiver and a receiving transceiver. The difference in path delays leads to interference between the originally received signal and delayed replicas of this signal. The hardware has the processing capability of distinguishing several of these replicas, provided they arrive within a given delay of each other, and coherently add their energy to improve the received signal energy. This capability is implemented by the multipath accumulator in the receiver circuit. The delay tolerable by this circuitry is approximately 6 microseconds when opearating at 100 Kbps rate, and about 2 microseconds when operating at 400 Kbps rate. The hardware can notify the processor when the multipath accumulator is in operation.

Bit error probability The noise process resulting from ambient background noise and receiver thermal noise are well modeled as Additive White Gaussian Noise (AWGN). This model corresponds to an ideal noise process which is stationary and has a constant-power spectral density accross all frequencies (white noise). The noise power is thus a linear function of the receiver bandwidth. When a correlation detector or matched filter detector is used to receive a signal[1], the noise energy registered by the detector in the absence of the signal is numerically equal to the value of noise-power spectral density. For detection in AWGN, with equally likely transmission bits, these detectors correspond to optimal detector design and lead to minimum bit error probability. They are also fairly straightforward to implement and a matched filter is used in the LPR hardware. The bit error probability with this detector, for a binary antipodal system like ours, is a function of the *signal-to-noise ratio*, SNR. The signal-to-noise ratio is defined as the ratio of signal energy to noise energy using an optimal detector, $\mathrm{SNR} \equiv P\ R.N_0$. In this expression P is the average received signal power, R is the channel bit rate, and N_0 '2 is the value of the AWGN spectral density[2]. The expression for the bit error probability is

$$P_b = Q\{\sqrt{\overline{SNR}}\} \quad . \tag{3.1}$$

[1]The detector is assumed to have a replica of the transmitted signal, normalized to unit-energy.

[2]This density refers to the double-sided spectrum convention

The function $Q(\cdot)$ is the Gaussian Q function. In practice, the MSK technique employed slightly improves the bit error probability, but $Q(\cdot)$ can be used as a good approximation to the bit error probability of an MSK system. A plot of this function appears in Figure 6.1. The error probability is independent of the signal shape. This implies that spread spectrum does not improve signal performance over Additive White Gaussian Noise, as long as the symbol energy is held constant.

The function in Figure 6.1 gives a one-to-one invertible mapping between signal-to-noise ratio and bit error probability. Thus we could estimate the signal-to-noise ratio in the channel based on an estimate of the bit error probability. The Parameter Selection uses an approximation to this function when trying to make bit error calculations. There is no known analytic inverse for the function, but the approximate function can be inverted readily. Bit error calculations are described in Section 4.2.

CDMA The use of pseudo-random chip sequences for spread-spectrum transmission is known as Direct-Sequence spread spectrum. When spread spectrum sequences, or codes, are used to provide multiple access capability, the system is known as Code Division Multiple Access (CDMA). In a CDMA system signal detection errors are due to channel noise (e.g. AWGN) and possible user interference. Spread spectrum sequences in a family of CDMA sequences are pseudo-orthogonal to each other and to time shifted versions of themsleves. In other words, overlapping spread spectrum signals look to each other like random noise. A detector matched to one sequence is poorly matched to other sequences and poorly matched to replicas of the same sequence which are time shifted by more than one chip period, modulo the spreading factor. When the receiver de-spreads the intended signal, by correlating the received signal with the local copy of the spread-spectrum sequence, it rejects most of the energy from signals of other users. The bit error probability in a CDMA system depends on the number of simultaneous users, their proximity and their power, the spreading factor, and the energy of the user signal being detected. Several authors have addressed the problem of characterizing this probability of error [11, 9]. It turns out that the same expression as for the AWGN case applies, but with the signal-to-noise ratio redefined. Let d_i denote the distance between the transmitter and receiver pair trying to achieve communication and d_j be the distance between and interfering transmitter j and the same receiver. Let o be the set of other nodes transmitting simultaneously within range of a receiver and P_j be the transmit power in use by interfering node j. Let T be the duration of a channel bit, assumed equal for all users. T is the reciprocal of the channel bit rate R. The signal-to-noise ratio for CDMA is defined as

$$\mathrm{SNR}_{\mathrm{CDMA}} \equiv \frac{P_i T}{N_0/2 + \sum_{j \in o} \left(\frac{d_i}{d_j}\right)^{\alpha} \frac{P_j T}{N}} \quad . \tag{3.2}$$

P_i is the received average power of the desired signal, from transmitter i. We have written the expression explicitly in terms of the transmitter powers, instead of just the received signal energy, so that we could incorporate the effect of near-far considerations, as in [9]. The summation in the denominator is over all interfering transmissions. The exponent α represents the propagation law that applies to the network. For isotropic propagation in free space $\alpha = 2$, since the spatial power density of an electromagnetic wave-front decreases with the square of the distance traveled. For propagation of radio signals along the ground, which is more realistic for packet radio networks, α is experimentally found to lie between 3 and 4. This value reflects the effect of multipath fading due to ground and obstacle reflections. Because of the multipath accumulators in the LPRs the effect is not expected to be as pronounced in the SURAP

environment. The propagation law implies that near radios contribute a lot more to interference than do far radios. An interfering radio at the same distance as transmitter i experiences the equivalent of having its transmitter power decreased by a factor equal to the spreading factor once the receiver de-spreads the signal. Differences in channel bit rates among different interfering users is immaterial since the chip rate is fixed. The spreading factor N in the expression represents the pseudo-orthogonality property of the spread spectrum sequences.

Equation (3.2) tells us that the effect of CDMA interference is equivalent to a time varying signal-to-noise ratio in the point-to-point channel between the transmitter and receiver attempting to communicate. It also says that CDMA interference is dependent on the channel parameters chosen by LPR's within range of the channel receiver. The fate of packets in this channel will lead to parameter adaptation, which may lead to the parameter adaptation in neighboring nodes. We will return to this issue later.

CDMA interferers An idea of the number of CDMA interferers that will overlap a packet reception at a given transceiver can be obtained by an analogy to the unslotted Aloha system. Consider the CDMA interference experienced by a test packet reception. Interference consists of packet transmissions, within range of the receiving transceiver, which are still in progress when the test packet arrives. It also consists of packet transmissions within range that begin while reception of the test packet is still in progress. Transmissions prior to arrival of the test packet must not have captured the transceiver, or the test packet will not be received, and are likely to be transmissions directed to other nodes. Transmissions made after the test packet arrives can be directed to the same transceiver as the test packet since the transceiver will have already locked onto the test packet. The process of interfering transmissions which are powerful enough to cause errors in the test packet is exactly analogous to an unslotted Aloha system In this Aloha system the collision probability is given by the probability that a transmission is present when the test packet arrives, times the probability that a transmission begins before the test packet ends. The throughput, S, of unslotted Aloha, $S = Ge^{-2G}$, peaks when the number of transmissions per mean packet time, G, reaches 0.5. The throughput at this point is $e^{-1} \simeq 0.18$ packets per packet time, for an average number of attempts per packet equal to $0.5/0.18 \simeq 2.78$. With the Poisson assumptions of the unslotted Aloha system, the maximum average of 1.5 transmissions per packet enforced by the Congestion Control Algorithm corresponds to $G \cdot S = 1.5 = e^{2G}$, or a number of transmissions per mean packet time approximately equal to 0.2. The probabilities of one, two, and three collisions within one mean packet time before or after the arrival of a test packet are 0.27, 0.05, and 0.007, respectively. We expect similar probabilities to apply in our network. The Retransmission Protocol we use will further reduce these probabilities since pure Aloha assumes a constant retransmission rate.

What the above example tries to argue is that situations in which packet receptions experience much more than 0.3 CDMA collisions on average are incompatible with a sound number of transmission attempts per packet in SURAP 4. Moreover, when other transmitters send to the same receiver, the number is even lower since 1.5 transmissions per packet must include the effect of transceiver blocking. The Retransmission Protocol will further reduce the number in either situation. The arguments discussed probably apply, with somewhat different values, to other network architectures as long as some congestion control mechanism is in place. High CDMA interference situations lead to a large number of retransmissions and flow control mechanisms throttle the affected flow to a point where CDMA interference stops being a problem. For this reason we prefer to treat CDMA interference as a secondary

consideration when adapting to channel no se and jamming. The noise perceived by the traffic, in steady state, will be mostly the channel noise and jamming noise in the channel, and only occasionally the CDMA interference. Realistic network simulations will decide if we need to modify the algorithm to overcome problems caused by CDMA interference.

Jamming Jamming is generally perceived as a decrease in channel signal-to-noise ratio. The amount of decrease depends on the specific type of jamming signal.

The original role of spread-spectrum communications was as an anti-jamming mechanism. The effect of Continuous Wave (CW) Wide-Band Jamming, which has a flat spectrum over the jammed signal bandwidth, is analogous to that of CDMA interference. The probability of error can be found by defining the signal-to-noise ratio as

$$\mathrm{SNR}_{\mathrm{CDMA,J}} \equiv \frac{P_i T}{N_0/2 + \sum_{j \in \phi} \left(\frac{d_i}{d_j}\right)^{\alpha} \frac{P_j T}{N} + \left(\frac{l_i}{d_J}\right)^{\alpha} \frac{P_J T}{N}} \quad . \tag{3.3}$$

where d_J is the jammer distance to the receiver. The jammer power appears attenuated by the spreading factor of N at the receiver. The result can be generalized to include situations with many jammers. The combined effect of jamming and CDMA interference is a time varying signal-to-noise ratio which depends on the number and power of interfering users, the power of the jammer, and the distances of users and of the jammer to the receiver. A similar expression applies to a narrow-band jammer, except that jamming power is attenuated by a factor less than N. For a pulsed jamming situation, where the pulse lasts longer than the channel bit, using the signal-to-noise ratio is usually not an appropriate way to compute the error probability and underestimates the proper value. Errors tend to cluster together and, for jammer power that is capable of producing bit errors, the error probability is mainly determined by the pulse duty factor. Clustered errors are overcome to some extent by the interleaving capabilities of the LPRs operating in FEC mode.

FEC In the SURAP environment a link implements FEC communication in the following form. A Cyclic Redundancy Check (CRC) field is computed for the packet to be sent and appended to it. The resulting packet is handed to the convolutional encoder in the LPR. The encoded packet is interleaved and transmitted. The receiver de-interleaves the packet and hands it to the sequential decoder. If the packet is decoded before a time-out interval the CRC is used to check if the decoded packet has any errors. In the event of time-out or packet errors the packet is dropped.

Characterizing bit error probability when FEC is used is not easy. Nachum Shacham [14] has analyzed the throughput achievable with ARQ strategies on a radio link using sequential decoding. The probability of packet error here was approximated by the probability that the sequential decoder times out before finishing decoding. This probability can be found by using the Pareto distribution approximation to the decoding delay of a sequential decoder. The probability depends on the time-out value chosen. The method could be extended to incorporate a model where packets decoded within the time-out period may still experience errors. The probability of these events would have to be incorporated into the model for packet error probability.

One way to represent the effect of FEC on bit error probability is as a coding gain on the signal-to-noise ratio. The same bit error probability after decoding, which is achieved by uncoded transmission,

can be achieved at a lower channel signal-to-noise ratio if FEC is in use. Alternatively, for a given channel signal-to-noise ratio, a lower bit error probability can be achieved with FEC than without it. We propose to represent the effect of FEC gain as a coding gain on the channel signal-to-noise ratio. That is, the bit error rate of the system is found by replacing $G^{FEC}P_tT$ for the desired signal energy P_tT in the expression for signal-to-noise ratio. The factor G^{FEC} depends on the FEC rate used and constitutes the coding gain on the signal-to-noise ratio. Unfortunately this gain will in fact depend on the channel signal-to-noise ratio. However, data from the Linkabit Corporation given to Hazeltine, the company in charge of LPR hardware development in the SURAN packet radio project, indicates that the concept of a constant FEC gain is a good approximation over the operating range illustrated in Figure 6.2. The figure shows the probability of bit error versus signal-to-noise ratio for several FEC choices using sequential decoding and using hard and soft-decoding techniques. In particular, the two curves in bold lines refer to rates 1/2 FEC hard-decoding and 3/4 FEC hard-decoding. The figure shows that the 1/2 FEC choices has an almost constant signal-to-noise ratio gain of 1.2 dB over the 3/4 FEC choice over the 10^{-4} to 10^{-7} bit error rate range. Moreover, the 3/4 FEC choice shows a signal-to-noise ratio gain that varies between 3 dB and 3.45 dB with respect to the bit error rate curve for binary PSK evaluated over the same bit error rate range shown in the figure.

The signal-to-noise ratio gain values obtained from Figure 6.2 differ from the gain values obtained from [3] and used in this report (see Table 3.2). We have used the values in [3] because at the time of obtaining the information the encoding-decoding hardware was still experiencing hardware problems and did not seem to be able to achieve the gains expected. Before implementation of the algorithm, it would be necessary to use field tested FEC performance curves to assign nominal FEC gains to each of the FEC coding rates.

Packet error probability Pursley and Taipale have also investigated the performance of convolutional codes, operating with a Viterbi decoder. For a packet of length L bits, the upperbound $1 - (1 - P_u)^L$ is derived for the packet error probability[3]. P_u is a bound on the probability of coming across the wrong decoding path in the decoder, and is independent of the packet length. The upperbound accurately describes the dependence of packet errors on packet length. Even though Viterbi decoding and sequential decoding are different mechanisms, the search procedure in sequential decoding is essentially a heuristic approach to the optimal search procedure of the Viterbi decoder. The role of P_u in the packet error probability of a sequential decoder is similar in nature and is often used as a first step in deriving bounds on the error probability of the decoder. We expect a similar dependence of packet error probability on packet length.

This type of dependence on packet length is the same, or similar in the case of sequential decoding, to the dependence that applies in an uncoded system. Under AWGN assumptions, a given channel signal-to-noise ratio translates into a bit error probability P_{bit}. Bit errors are independent, identically distributed Bernoulli random variables and the probability of a packet reception containing one or more bits is

$$1 - (1 - P_{bit})^L \tag{3.4}$$

The dependence of (post-decoding) packet error probability on packet length will be exploited by the algorithm in the context of error acknowledgements (Section 4.2).

[3] P_u is the union bound for first-event error probability.

Table 3.2: SNRGain Value

Power	gain (dB)	FEC	gain (dB)	Channel rate	gain (dB)
20 mW	0	none	0	400 Kbps	0
125 mW	8	7/8	?	100 Kbps	6
800 mW	16	3/4	4.6		
5 W	24	1/2	7.5		

Gain The SNRgain obtained by increasing power or decreasing channel bit rate is due to the increase in symbol energy P_tT. The gain in these cases is a fairly robust parameter and is easy to calculate. The gain obtained from decreasing the FEC rate is a coding gain. It depends on the channel noise process and in particular on the distribution of channel bit errors. Interleaving, for example, tries to ensure that errors are not clustered, even in pulsed jammer situations. FEC gain also depends on the prevailing signal-to-noise ratio value in the channel. It is a less robust, less predictable, quantity than gain resulting from symbol energy. A nominal FEC gain may be calibrated for the LPR's for ideal AWGN channels. Under certain situations, however, the actual gain experienced may be less than the nominal gain. The algorithm design must take into account this possibility to avoid implementing a gain lower than the gain required by the channel.

Table 3.2 expresses the information of Table 3.1 in the form of relative SNRgain with respect to a set of reference parameters. The reference parameters chosen are lowest power, no FEC, and 400 Kbps channel bit rate. They are chosen because each of them gives the least amount of protection against noise among all possible parameters. The information on coding gain for FEC values was obtained from [3] and corresponds to the values for hard-decoding. The gain value for 7/8 FEC rate was not available. We will use a nominal value of 4 dB for this gain when a numerical value is necessary in the report. The nominal gain for a choice of parameters, relative to a choice of lowest power, no FEC, and 400 Kbps, can be found by adding the individual relative gain of each parameter using the values in Table 3.2. Consider a gain comparison between two different parameters. Since the true gain of the FEC choice can turn out to be different from the nominal gain in the table, we cannot be certain of the result of the comparison in some cases. As an example consider the following two sets of parameters. The first choice consists of 800 mW of power, FEC rate equal to 1/2, and channel bit rate of 400 Kbps. The nominal relative gain is 23.5 dBs. The second choice consists of 800 mW of power, no FEC, and channel bit rate of 100 Kbps. The nominal relative gain in this case is 22.0 dBs. The conclusion is that the first choice gives greater gain and hence should perform better in noise. However, suppose that the noise consists of pulsed and chirped jamming, and that this results in a true relative gain of only 5 dBs for FEC rate 1/2. Then the first choice of parameters has a true relative gain of 21.0 dBs, and will in fact perform worse than the second choice.

This problem in deciding which parameter triplet gives greatest gain is a result of the limited model we are using for FEC gain. However, modeling FEC gain for sequential decoding over the whole range of operating conditions is difficult. To live with the problem one could simply use a random search over the set of possible choices until the one that performs best is found. Unfortunately this could compromise the speed at which the algorithm adapts to changes in noise. Our approach consists of using

a set of parameter choices for which we are certain of their relative gains. This is achieved by using a set of parameter combinations so that if the nominal gain of one combination is greater than the gain of another combination, each parameter in the first combination has a gain greater than the corresponding parameter in the second combination. The combinations in the example given above do not fulfill this parameter-wise monotonicity property and hence these combinations would not be used by the algorithm.

3.2 Contention and CDMA Interference

The following factors prevent successful packet reception:

- Blocking, i.e. the receiving transceiver is busy
- CDMA interference resulting in bit errors
- Channel noise resulting in bit errors

We assume that if a packet arrives while the transceiver is idle, the transceiver can synchronize to the packet and reception begins. CDMA interference is related to both blocking and channel noise. It is related to blocking in that it correlates with channel contention: it occurs only when packets overlap at the receiving transceiver. It is related to channel noise in that, given there is overlap, it leads to packet errors in a manner similar to channel noise by decreasing the signal-to-noise ratio. In one fundamental sense, however, CDMA interference is better treated as analogous to blocking than to channel noise. Channel noise levels are independent of network behavior while CDMA interference, like blocking, is affected and can be controlled by network behavior. The Parameter Selection Algorithm is the only mechanism the network has to adapt to channel noise. Because link-layer protocols and end-to-end flow control regulate blocking, and hence CDMA interference, our design is biased towards solving the problem of link quality in varying channel noise. CDMA interference is treated as a second order problem. We will return to this issue later in this subsection.

It is important for the algorithm to distinguish if attempted transmissions fail because of blocking or channel noise. In contention situations the appropriate action is to minimize packet overlap at the transceiver, and to minimize transmission power in the case of CDMA interference. These goals can be accomplished by reducing the rate of packet transmissions in the neighborhood of the receiving node and/or the packet overhead due to coding and channel bit rate. In the second situation the appropriate action is to increase channel gain. This would entail increasing power, decreasing FEC rate, or decreasing channel bit rate, all of which run counter to the measures required in blocking and CDMA interference.

Contention Traffic rates and packet durations on the channel are the factors responsible for packet overlap at a receiving node. The probability of a packet being blocked will monotonically increase in both of these factors. A packet sent to a node faces contention for the transceiver from other transmissions to the same node, and from transmissions by that node to other neighbors. In particular, let λ_j denote the carried load, in packets per unit time, from a neighbor j sending to the receiving node and λ'_k be the carried load from the receiving node to a neighbor k. Carried load means the steady state sustainable packet throughput rate. If the average number of transmissions per packet on an output link k is r'_k, and

the duration of a packet on a link k is t_k, then the probability of packet blocking at the receiving node can be assumed monotonically increasing in

$$\sum_{\Theta} \lambda_j t_j + \sum_{\Lambda} \lambda'_k r'_k t_k \quad . \tag{3.5}$$

where Θ denotes the set of other nodes transmitting to the receiving node and Λ denotes the set of nodes that this receiving node transmits to.

The network has two main mechanisms to regulate the level of contention at network nodes. These are the link-layer retransmission strategy and the end-to-end flow control algorithm which are documented in [8] and [5] respectively, and were summarized in Section 2. The retransmission strategy is akin to a polynomial back-off strategy. The more attempts required for a packet transmission the longer the wait is to the packet's next retransmission. The wait is a constant plus a random part over an interval which is an affine function (linear plus constant) in the number of attempts. Thus, in a situation with many nodes contending for a transceiver, the strategy attempts to prevent a retransmission avalanche. The end-to-end flow control algorithm is part of the Congestion Control Algorithm. When the number of transmission attempts per packet to a given node exceeds a "target" value, the receiving node will cause participating flows to slow down by reducing their flow allowance (path ration, in the jargon of the Congestion Control Algorithm). Increasing channel gain in situations like this would have the adverse effect of introducing unwanted power or channel packet time overhead, which only increases contention or CDMA interference.

CDMA Interference SNRgain increases CDMA interference. CDMA interference among nodes sharing a common receiver node can be controlled, to some extent, by link-layer and end-to-end flow control actions. Interference resulting from traffic directed to other receiver nodes cannot be directly controlled by these network actions and is likely to result in a certain amount of gain increase.

The probability of CDMA interference is similarly related to traffic rates and packet durations in the vicinity of the receiving node. This time however, the probability includes not only the events consisting of transmissions in progress when a packet arrives at the receiver, but also events consisting of transmissions within range that begin while the reception of this packet is in progress. Let Ψ be the set of radios within range of the receiving node which transmit to nodes other than the receiver of interest. The probability of a transmission overlapping reception of a packet from neighbor i at this receiver can be assumed proportional to the value of the expression in (3.6). For convenience we have separated the terms into two groups. The first group represents traffic from the set Θ, and would be directly affected by flow control actions from the test receiver. The second group represents traffic from the set Ψ, and would only be affected indirectly by flow control actions from the same receiver. The constants a_j are introduced in the expression to account for the near-far CDMA phenomenon. They equal one if the transmission on link j is capable of causing measurable interference at the receiving node and equal zero otherwise. This includes the distance and spreading factor discussed in subsection 3.1.

$$\sum_{\Theta} a_j \lambda_j r_j (t_j + t_i) + \sum_{\Psi} a_k \lambda_k r_k (t_k + t_i) \tag{3.6}$$

The correlation that exists between blocking and CDMA interference can be seen in this expression. The flows from neighbors j sending to the receiving node under study, and the packet durations t_j for these

flows, increase both blocking and CDMA interference. The retransmission strategy, in the short run, and the flow control, in the long run, will throttle traffic until the average number of transmissions per packet to the node stay below the target value, currently set at 1.5. Flows corresponding to transmissions not directed at this receiving node are, to a first order, unaffected by the network control actions described. Hence they remain and are registered as background noise by nearby nodes. If their interference results in a completely lost packet transmission, the retransmission strategy and the flow control will tend to adjust to it. This is the case for nodes very close to each other. If their interference results in occasional bit errors, the Parameter Selection Algorithm will perceive them as channel noise and increase channel gain. This is a risky action since two nodes within range can lock themselves into a futile battle of increasing channel gains. Since there is an element of randomness in packet transmissions, and hence packet overlap, and since the retransmission strategy comes into play on failed attempts we have considered this possibility a second-order consideration. We are in the process of modeling the behavior of the algorithm in a realistic simulator that includes all link-layer protocols and a detailed model of the radio channel. These simulations will help decide to what extent the above concern is a problem. Alternate Routing Protocol presents another consideration because it may also contribute to increased transmissions in the neighborhood, creating a tendency to increase channel gains unnecessarily.

Contention versus Noise We have simplified the problem of how to adjust channel gain to a problem of distinguishing channel contention from channel noise and reacting accordingly. It is appropriate to ask what the strategy should be for the Parameter Selection Algorithm in a given network. When a transmission fails because of channel contention, the Parameter Selection Algorithm should not respond by increasing gain because the joint action by nodes transmitting to a common transceiver will probably result in increased contention or CDMA interference. Another interesting possibility is to require the Parameter Selection Algorithm to decrease the gain in this situation. This rule is being tested in the version of the algorithm developed by Rockwell, a contractor for the SURAP 3. The idea is to reduce the contention process. We have preferred to leave the responsibility of dealing with contention to the link-layer and network-layer algorithms. The back-off strategy and a stable feasible flow operating point seem better suited to control contention and CDMA interference than the Parameter Selection Algorithm, whose primary goal is to adapt to noise.

Now consider a situation where channel noise is the cause of failed transmissions. We look at the channel packet time overhead, and the packet forwarding delay on a link, as a function of a reference channel signal-to-noise ratio, for several combinations of parameters. We will argue that a compromise between high link throughput and low channel overhead can be achieved by maintaining a low value of packet error probability. This analysis considers a situation with no contention on the link. All retransmissions stem from packet errors because of noise. The combination of parameters presented is based on the desire for monotonicity of gain over an ordered set of parameter selections (see Section 4.2). Even though our algorithm design calls for always using some FEC, we have included a "no FEC" selection in the analysis, and excluded the rate 7/8 FEC parameter choice. This FEC rate was excluded from the analysis because there was no nominal gain available for this FEC choice. The method to be explained is general enough that these considerations will not alter the conclusions.

Figure 6.3 plots the average packet-time overhead in the channel as a function of channel signal-to-noise ratio. The plots correspond to 1000-bit packets. The overhead is normalized to a 1000-bit packet

transmitted at 400 Kbps and no FEC, this is, a 2.5 millisecond transmission. The overhead includes retransmissions resulting from packet errors. Selection 1 corresponds to no FEC, 400 Kbps; selection 2 is 3/4 FEC, 400 Kbps; selection 3 is 1/2 FEC, 400 Kbps; selection 4 is 1/2 FEC, 100 Kbps.

To be able to compare their performance they are all ploted against the SNRat the receiver that would apply for selection 1. The overhead is found by computing the packet error probability, P_{pkt}, for the SNRand computing the expected value of channel use in transmitting the packet. P_{pkt} is found by adding the relative channel gain of the selection with respect to selection 1, expressed in dB's, to the SNRvalue experienced by selection 1. The resulting higher SNRvalue is used as the SNRvalue in Figure 6.1 to find the bit error probability and then equation (3.4) to compute P_{pkt}.

For each selection it can be seen that for sufficiently high SNRvalues, the overhead corresponds to a single transmission of the packet. For sufficiently low SNRvalues, the overhead corresponds to six transmissions of the packet. The region in between corresponds to P_{pkt} varying between the two extremes $P_{pkt} \simeq 0.0$ and $P_{pkt} \simeq 1.0$. The slopes of the transition reflect the individual packet overhead that applies for each selection. To minimize channel overhead, the algorithm should strive to operate along the hull of the combined loci of curves in the graph. In theory, as channel signal-to-noise ratio decreases, the algorithm should change from selection 1 through 4, in numerical order. The optimal switch-over points in the figure correspond to SNRvalues of 7.7 dB, 2.9 dB, and -1.23 dB in the horizontal axis. The packet error probabilities that apply can also be computed. While using selection 1, P_{pkt} increases from 0.0 to 0.24, when the change to selection 2 occurs. With the new selection the probability decreases again to near zero. As SNRdecreases further, the change to selection 3 occurs when $P_{pkt} \simeq 0.33$, and to selection 4 when $P_{pkt} \simeq 0.84$. If SNRincreases, P_{pkt} will tend to zero, and the algorithm should attempt to switch back progressively towards selection 1.

Having considered optimal parameter selections for minimizing overhead on the channel, let us consider optimal selections for minimizing the packet forwarding delay, still assuming no channel contention. Figure 6.4 shows the average packet forwarding delay for the same system of 1000-bit packets. The delay is normalized to 2.5 milliseconds as before. The delay is computed by computing P_{pkt} for the SNRvalue and computing the expected total delay to packet success using equation (2.1) of Section 2. Link throughput can be found as the reciprocal of link delay. For each selection we observe that for high SNR values the delay consists simply of the transmission time. At a certain SNR value there is a steep increase in delay with decreasing SNR. This is due to retransmissions. To maximize link throughput, or minimize link forwarding delay, the algorithm should strive to operate at the hull of the loci, as for minimizing overhead, except that the switch-over points have changed. The SNRvalues obtained as the algorithm switches from selection 1 through 4 are 9.55 dB, 4.65 dB, and 0.62 dB. The corresponding values of P_{pkt}, for the selection prior to each transition, are 0.01, 0.02, and 0.15, respectively. As can be seen from the graph, and shown by these values, the switching must take place as soon as the probability of a second packet attempt becomes significant.

We can now make the following observations.

1. When the cause of a failed transmission is channel contention increasing gain worsens the situation.

2. There is a trade-off between optimizing channel overhead and optimizing link throughput, in the absence of contention.

3. Beyond small values of packet error probabilities, the link throughput is very sensitive to changes

in this probability.

4. Changes in channel overhead or link throughput are small to moderate when changing between selections 1 and 2, and between selections 2 and 3.

The observations in this section lead to the following algorithm. If the Parameter Selection Algorithm believes that the cause of an unsuccessful transmission is contention, it should refrain from increasing gain. We prefer not to decrease gain to leave the responsibility of controlling contention to higher-level algorithms. When facing noise, the algorithm should use the selection of parameters with the lowest possible overhead, so that the packet error probability does not exceed a value FAIL. This is a heuristic compromise between minimizing overhead and maximizing link throughput. It has the simplicity that packet error probabilities are, to some extent, readily observable. It also has the simplicity of using the same threshold value on P_{pkt} for all transitions. Since channel overhead relates to blocking probability and CDMA interference, and since flow control mechanisms in the network can address these problems to some extent, we prefer to bias the value of FAIL towards maintaining good link throughput. This also prevents link delays from exercising involuntary flow control action, which may well disrupt the fair flow control allocations of the Congestion Control Algorithm.

We have not included a characterization of optimal operation regarding transmitter power levels. Since the choice of power will determine packet error probability the link throughput considerations still apply intact. Instead of channel overhead, however, it is CDMA interference we are concerned about this time. In previous sections we have argued that they correlate to blocking events and that network flow control can assist in this respect. Hence we expect CDMA interference considerations to be similar to channel overhead considerations and the same value of FAIL is used for determining all transitions between selections, whether these consist of power changes, FEC, or channel bit rate.

We recommend a value of FAIL = 0.1 for SURAP 4. The channel overhead experienced prior to transitions, as the gain is increased from selection 1 to 4, has values 1.1, 1.5, and 2.2, in that order. The corresponding values of average forwarding delay are 4.6, 5.0, and 6.0. Thus we can argue that the physical link layer will show a fairly stable profile to higher-layer algorithms as it switches up to selection 4. Even the overhead and link delay once selection 4 is chosen do not seem too disruptive, especially given the alternative of loosing the link.

A retransmission strategy with no back-off, i.e. where successive retransmission waits are similar in magnitude, would lead to a different operating point, with a higher value of FAIL. However one other consideration also points to maintaining a low value for packet error probability in noise: packets are discarded after a finite number of retransmissions. In the SURAP architectures the maximum number of packet attempts allowed is six. A high value of packet error probability, combined with the operating values of packet blocking probability, could lead to an unacceptably high average number of retransmissions and packet drops.

4. Algorithm Design

4.1 Design Principles

In an unrestricted sense the objective of the Parameter Selection Algorithm should be to choose the right combination of channel parameters for each directed link in the network, so that desired end-to-end flows in the network remain fair, network throughput is maximized (possibly subject to delay constraints), and connectivity in the network is robust. The algorithm should adapt quickly to changes in the network radio environment. An optimization of this kind would require global network-state information, more complete models of channel interactions, and probably a large amount of computation. We have chosen to take a local view of the problem optimizing the parameters for a given link and its adjacent links. The motivation behind these design principles was discussed in detail in Section 3. Subsection 4.2 contains a detailed description of the algorithm.

Below is a list of the main principles followed in the design of the Parameter Selection Algorithm:

1. Work with a local solution instead of a global solution to simplify design.
2. Maintain network connectivity, using maximum gain on a per-packet basis if necessary, to achieve survivability.
3. Distinguish between contention situations and channel noise, and focus on adapting to noise.
4. Maintain a low value of packet error probability to guarantee acceptable link throughput and provide a stable physical layer.
5. Minimize the impact of transmissions on adjacent links.
6. Adapt quickly, using bit error statistics if possible.
7. Provide fairness by guaranteeing acceptable packet error probabilities even for maximum-length packets, which are more prone to errors.
8. Recover from estimation and decision errors to provide robustness.

Local Solution To simplify the design problem we implemented a local solution, as opposed to a global solution, to parameter selection. At each link, individually, the algorithm tries to minimize link gain, hence its adverse impact on the link's neighbors, subject to maintaining a robust link. The solution is local in the sense that it ignores the possibility of cooperating among nodes to implement a joint solution which optimizes the operation of the network as a whole. The simplification avoids the need for sharing global state information and the associated computation of the optimal solution. Perhaps the biggest

obstacle to a global solution is the lack of detailed models of channel interactions among nodes and of the interaction among the different network algorithms.

Connectivity Connectivity is vital to the survival of a network. Haphazard connectivity causes frequent routing updates, which are likely to degrade the performance of the network. Ruptured links must be detected quickly to prevent congesting the network with doomed packets.

The algorithm increases the channel gain as the maximum number of attempts permissible is approached. Maximum gain is always used on the sixth transmission attempt. This implies that a link is maintained by the algorithm, on a packet by packet basis, even if it requires transmissions at maximum gain, unless the maximum number of neighbors is exceeded or unless the link up/down protocols decide that the link requires too many attempts per packet on average. It also implies that a ruptured link is detected within approximately one packet-forwarding time provided there is traffic flowing through the link.

The SURAP architecture requires periodic broadcasts from each node of special packets containing network information. They are called PROP packets and are used to start the process that establishes a link and maintains a live link in the absence of traffic. We require that PROPs be broadcast at maximum channel gain at least periodically.

Contention versus Noise It is important to distinguish between channel contention and channel noise because the right reaction to one situation tends to be the wrong reaction for the other. A packet radio network is a multiple-access environment, which is designed to experience channel contention even during normal steady state operation. Higher-layer algorithms in the network are designed to keep this contention in check.

The algorithm follows two approaches to differentiating between contention and noise. In the absence of any reply after a data transmission, the algorithm assumes that the cause is contention for the first two attempts, and that the cause is noise during the next three attempts. This approach is equivalent to a heuristic hypothesis test, where the hypothesis explaining the event "no reply" goes from contention to noise as the number of attempts increases. The hypothesis is changed after three transmission attempts because theoretical and experimental studies ([17, 5]) indicate that more than 1.5 transmissions attempts per packet on average are near the limit value of contention, before congestion sets in. Since we consider it important to let other algorithms deal with contention we consider three transmissions a safe margin.

The second approach is the use of error acknowledgements, or *whacks*. When discussing packet error probability in Section 3.1, we saw that the probability of one or more bits in error within a given field of bits inside a packet increases with the length of the field. We expect that we will often be able to determine who sent a packet even if the decoded packet contains errors. A whack, returned to the sender as a regular ack packet, would be positive proof that the cause for failure is noise and not contention. The remaining ambiguity between channel noise and CDMA interference is tolerated for the reasons discussed in Section 3.

Low Packet Error Probability Packet error probability here refers to the probability of a packet failing the CRC check at the receiving node, provided that the packet arrives without being blocked. In Section 3.2, when discussing contention versus noise, we saw that optimal operation for link throughput

required relatively low values of packet error probability, and that higher values led to a steep rise in link forwarding delay. Link throughput is our measure of the quality of a link, and packet error probability is a convenient way to express it. The algorithm maintains a packet error probability below a value FAIL, set to 0.1 for SURAP 4, by increasing the gain until a successful transmission is achieved, and not decreasing that gain until at least $FAIL^{-1}$ consecutive successes are experienced. Bit error calculations will also assist this process and are discussed below.

Allowing large forwarding delays on a link to minimize the CDMA interference caused by the link, or to reduce blocking probabilities for transmissions directed to the same receiving node, would present two problems. The first is the reduction in the link's throughput. It is not clear that the throughput of other links should have priority over this link's throughput. The second problem relates to fair network flow. If traffic in a link slows down at the link level, the throttling action on the flows sharing the link gives no regard to the end-to-end fairness consequences that result. If traffic slows down in the link, other links sending to the same node experience less contention which may lead them to increase their flow and further penalize this link. These actions run counter to the concept of Max-Min Fair flow allocation. If instead the link kept approximately the same traffic rate, by increasing its gain and contributing to contention, the problems in the link would be shared by everyone transmitting to the node, and the Congestion Control Algorithm would adjust all the flows accordingly, in a fairer manner.

Minimize Contention If a given gain state is capable of maintaining the packet error probability in a link below a threshold value, so can a gain state of higher gain. However, the latter state causes greater blocking and CDMA interference to nearby radios than the former. Hence we ask the algorithm to minimize the gain subject to guaranteeing a robust link, as expressed by FAIL. The algorithm achieves this by trying to decrease its gain when it believes that the packet error probability must be equal to or less than the threshold. If the transmission after decreasing the gain is successful, the algorithm is one gain state closer to the minimum. If this new gain state leads to errors instead, the algorithm soon increases the gain, typically right after the failure. In this fashion, the average packet error probability will be maintained around its threshold value after several cycles of decreasing and increasing gains. This means that if channel noise changes, the algorithm adapts and eventually settles around the gain states that have gains just above and just below the necessary gain. This process achieves the constrained minimization of contention desired.

Quick Adapting The speed at which the algorithm adapts presents a problem. The algorithm does not know the channel characteristics, but rather estimates them from channel feedback. Smoothing out the uncertainty involved in this estimation process requires the algorithm to delay decisions about changes in channel gain. Fast adaptation, on the other hand, calls for quickness of decisions based on this feedback.

The algorithm can adapt to increases in noise fairly soon, sometimes increasing one gain state per failure, after receiving whacks. It decreases the gain after $FAIL^{-1}$ successes, and so adapts to decreases in noise level at a rate proportional to $FAIL^{-1}$. Typically steady state behavior will be $FAIL^{-1}$ successes followed by one failure, for a packet error rate of FAIL. Bit error rate estimates also improve the speed at which the algorithm adapts.

With the packet sizes present in the network, and with FEC in use, large sample sizes of channel bits are possible even with one packet. A 1000-bit packet, using a rate of 1/2 FEC provides 2000 channel

bits. If the packets are decoded correctly, the sequential decoder can say with some certainty how many channel-bit errors were present by comparing the channel bits of the decoded packet with the channel bits received. Assuming the noise is stationary, we are able to determine with some degree of accuracy the bit error rate in the channel. This can be used to compute a decrease or an increase in gain. The fact that we can successfully transmit a packet and yet consider that we are using insufficient gain after calculating its channel-bit error rate are not incompatible because the success of a packet is a statistical event. We will return to this point in Section 5 which discusses performance. The algorithm accumulates bit error statistics of this type to assist in making decisions to increase and decrease gain. The exact way in which this is done is described in subsection 4.2.

Fairness One aspect of fairness has already been mentioned when discussing connectivity. A link should not be penalized for requiring a large gain. The algorithm tries to maintain the same link quality even if maximum gain is required. A related aspect was mentioned when discussing packet error probability. If the link throughput is allowed to deteriorate for the sake of reducing the contention and interference caused, the algorithm may end up working at cross-purposes with the Congestion Control algorithm and may jeopardize Max-Min Fairness. There is one more fairness consideration. For a stationary AWGN channel, the packet error probability is an increasing function of the packet length. For a given gain, short packets are less prone to errors than large packets. Hence, when calculating the gain required, we ask the algorithm to calculate the gain required to maintain a robust link for traffic of maximum-length packets. The implementation of this is simple when bit error statistics are available. The algorithm knows the signal-to-noise ratio needed for a maximum-lenght packet to have a packet error probability no greater than FAIL, by using the information in Figure 6.1. It can translate the current bit error rate to a signal-to-noise ratio by using the same figure to compute the change in gain required. Notice that the size of the packets that provided the bit error statistics is not relevant. Operating at this gain guarantees that if occassionally large packets travel through the link, they will not experience poor link quality even if regular traffic through the link consists of small packets. In general the probability of packet error is assumed of the form $1 - (1 - p)^L \simeq Lp$. p is dependent on the signal-to-noise ratio and L is the packet length. The approximation applies when p is small. This implies that a factor of ten difference between packet sizes will lead to a factor of ten difference in packet error rates. For this reason the fairness consideration is relevant. We would not correct for this bias when bit error statistics are not available (no FEC) for simplicity, since the implementation in this case is not as straightforward.

Recover from Errors Errors of estimation occur when an unlikely event occurs, for example ten successful packets in succession when using a very poor link. Decision errors occur when the calculations are erroneous even if the data is correct, for example if the bit error rate estimated is accurate but the nominal gain of an FEC choice overestimates the gain in the face of a specific jamming signal. It is important that the algorithm recovers from such errors when they occur so that it does not perpetuate the wrong choice of gain.

Two elements in the algorithm allow it to recover from errors. The first is the use of a monotonic gain table whose gain entries are given as differential gain. This means that at each new gain state the algorithm re-estimates the quality of the link and if it is unsatisfactory it corrects it in the right direction by looking at the relative change in gain as it moves up and down the gain states in the table. Moving

in the "right direction" is insured by the monotonicity of gain in the table entries. The change in state is insured by the fact that, since the algorithm does not work with absolute gain values, the algorithm is not fooled into thinking that "it already has enough gain." The second element is the fact that the algorithm is biased towards believing observations of packet failures and successes, over estimates of bit error rates. This means that regardless of the reasons for packet errors, the algorithm will tend to adjust gain until it restores the link quality desired, if at all possible. Bit error estimates are believed over packet error estimates only when it is fairly certain that a packet error will follow otherwise. In general bit error calculations lead to gain state transitions only to dictate more gain than the gain determined based on packet observations only. The details are discussed in detail under sections 4.2 and 5.

4.2 Algorithm

This subsection describes the Parameter Selection Algorithm in detail. We divide the algorithm into seven elements, which are described below. The sixth element, Link Up/Down Protocol, is included here because it is closely related to the Parameter Selection Algorithm, but could also be regarded as a separate algorithm. The elements are:

1. Link State
2. Feedback Set
3. Transition Rules
4. Bit Error Computation
5. Ack Mode
6. Link Up/Down Protocol
7. Packet Format

Link State The Parameter Selection Algorithm runs on each node of the network. Each directed link in the network is maintained individually. A node i keeps a *link state* for each outgoing link (i, j) to a neighbor j. The algorithm adapts the channel gain for each output link individually based on the link state and the observed replies from the neighbor j. A link state is determined by the value of five variables:

1. *gain_state*
2. *tx_count*
3. *ack_count*
4. *bit_count*
5. *bit_errors*

The gain state variable *gain_state* is an integer which identifies a triplet (P, FEC, R). The triplet is the parameter selection. P, FEC, R denote power, FEC rate, and channel bit rate. The possible selections are kept in a special table called the *gain table*. The table is an ordered set of selections. The position in the table is the gain state. The table has two important properties. First, the SNR gain of the selection is monotonically increasing in *gain_state* . Second, the channel overhead of the selection is monotonically increasing in *gain_state* . Channel overhead refers to the combination of power in use and of packet time overhead. If gain state k is greater than gain state l then $P_k \geq P_l$, $FEC_k \geq FEC_l$, $R_k \geq R_l$. The subscripts identify the state. The gain table associates a *differential gain* ΔG, expressed in dBs, to each gain state. The differential gain for a gain state is pre-computed by calculating the nominal relative SNR gain of each selection from Table 3.2 and subtracting it from the relative gain of gain state 0. The relative gain for a selection is the addition of the relative gains shown in Table 3.2 for each parameter in the selection.

Table 4.1: SURAP 4 Gain Table

gain_state	ΔG (dB)	**selection**
0	0.0	(20 mW, 7/8, 400 kbps)
1	8.0	(125 mW , 7/8, 400 Kbps)
2	8.0	(800 mW , 7/8, 400 Kbps)
3	8.0	(5 W , 7/8, 400 Kbps)
4	0.6	(5 W , 3/4, 400 Kbps)
5	2.9	(5 W , 1/2, 400 Kbps)
6	6.0	(5 W , 1/2, 100 Kbps)

The gain table recommended for SURAP 4 appears in Table 4.1. We have assigned a nominal value of 4.0 dB coding gain to the FEC rate 7/8 to be able to compute the table. (The measured value is not available.) All selections use hard decoding.

The *transmission count* variable *tx_count* is the number of transmissions attempted for the current data packet being forwarded on the link. It runs from 0 to 5. The *acknowledgment count* variable *ack_count* is the number of consecutive packets acknowledged since the last gain state transition[1]. This includes situations where the ack arrives on a second or third attempt, if earlier attempts had no reply. The *bit count* variable *bit_count* is the number of channel bits received by neighbor j, reported in its acknowledgments. The *bit errors* variable *bit_errors* is the number of channel bits received in error by neighbor j, reported in its acknowledgments. The *data cycle* variable *data_cycle* is a binary variable which takes the value ON if an unacknowledged data transmission to j is pending in the directed link (i, j), and takes the value OFF the first time that an ack for the pending packet is received. This data cycle variable acts as an enable/disable variable for the *ack mode*. This mode is explained below in detail.

All link state variables, except the gain state variable, are reset to zero after a gain state transition. The operation of the Parameter Selection Algorithm can be visualized as a walk up and down the gain states in the gain table. The order and number of the entries in the table is decided by the designer. As long as the table satisfies the monotonicity property, the algorithm is blind to the choice of entries.

Feedback Set Feedback is the outcome of observing the directed (j, i) link after a transmission on the directed (i, j) link. The reverse link (j, i) is the *feedback channel*. The Parameter Selection Algorithm recognizes three types of feedback: $0, e, 1$. Thus the outcome of the feedback channel is a ternary feedback set.

Feedback 0 refers to no reply on the feedback channel. It corresponds to the expiration of the retransmission timer in link (i, j) before an ack, a whack, or a nack are received for the pending transmission. We assume that only four events result in feedback 0: transmission on (i, j) was blocked, transmission on (i, j) was unrecognizable because of errors, the reply on (j, i) was blocked at i, the reply on (j, i) was unrecognizable because of errors. Even though the fate of the channels (i, j) and (j, i) are corre-

[1] As will be seen later self transitions are valid transitions

Table 4.2: Feedback Set

Feedback Type	Event
0	rtx time out
e	whack
1	ack, nack

lated, since each is the feedback channel for the other, these channels are maintained separately, by their sending node. Because of the Moment of Silence Algorithm, and the small length of acks, whacks, and nacks, problems in the feedback channel are assumed to have lower probability than problems in the data channel. Hence feedback 0 is interpreted by a node as the possible result of blocking or noise in the (i, j) channel. The ambiguity is nominally solved by looking at variable *tx_count* in the link state. When $0 \leq tx\_count \leq 1$, blocking is considered the cause. When $2 \leq tx\_count \leq 4$, noise is considered the cause. This was discussed in Subsection 4.1.

Feedback ϵ refers to the reception of a whack on the feedback channel. It identifies a noise error in channel (i, j). As discussed in Section 3, this noise may include CDMA interference but the algorithm reacts as if background noise were the cause of errors. A whack packet can also be a nack. This corresponds to a situation where neighbor j received the packet with errors, but would have nacked the packet anyway had it been received with no errors.

Feedback 1 refers to the reception of an ack, or a nack, on the feedback channel. It unambiguously identifies a successful reception on the channel (i, j). When FEC is in use, an ack or nack packet will also contain the number of channel bits received by neighbor j, and the bit error count calculated by its sequential decoder. Whacks do not contain this information since the calculations by the sequential decoder should not be trusted when decoding errors occur.

Table 4.2 depicts the feedback set used by the Parameter Selection Algorithm. A data transmission will always result in one of the three feedback types.

Transition Rules Whenever feedback is received on link (j, i) the link state is updated in link (i, j) and checked for gain state transition. If the feedback is of type 1, *ack_count* is incremented by one and *bit_count*, *bit_errors* are incremented by the amount written into the ack packet by neighbor j. All link state variables, except the gain state and the data cycle, are reset to zero after a gain state transition. *data_cycle* is set to ON at the first transmission of a data packet, and is reset to OFF the first time feedback 1 is received for the packet. A set of *transition rules* govern the gain state transitions.

A *self transition* is a gain state transition which results in the same gain state. The transition rules are independent of the current gain state in the link. A self transition at the highest gain state occurs when the transition rules determine a need for gain increase. A self transition at the lowest gain state occurs when the transition rules determine a need for a gain decrease. A self transition at an intermediate gain state occurs if FAIL uninterrupted successes occur but bit error statistics advise against decreasing the gain. Self transitions also reset the link state.

If bit error statistics are available, the algorithm computes the *bit error rate* variable *bit_error_rate* = *bit_errors*/*bit_count*. This variable and the gain state, or more precisely the parameter selection, are used

to estimate two things: first, the current quality of the link, and second, the change in gain required to achive acceptable quality. The calculations are described further below, but the transition rules depend on the result of these calculations.

Transition rules specify two things, the direction for gain-state change and the amount of gain state change. The outcome is a function of the link state and the feedback received. What follows are the transition rules categorized by the type of feedback received. *min_gain_state* denotes the lowest gain state. We will use the following notation:

$$\begin{aligned} M &\equiv \text{maximum gain state} \\ L &\equiv \text{minimum gain state} \\ k &\equiv \textit{tx\_count} \\ i &\equiv \textit{gain\_state} \\ j &\equiv \textit{new\_state} \end{aligned}$$

gain_state and *new_state* denote the current gain state and the gain state after transition.

0 If $0 \leq tx\_count \leq 1$ make no transition. If $2 \leq tx\_count \leq 4$ make a transition to gain state

$$j = i + \lceil (M - i)\ (5 - k) \rceil \qquad (4.1)$$

Equation (4.1) says that if *tx_count* =2, the next state is higher than the current one by one third of the difference between the maximum gain state and the current state, if *tx_count* =3 then by one half, if *tx_count* =4 then by the full distance. This guarantees a fast increase up to maximum gain in the last attempt.

e If $0 \leq tx\_count \leq 1$ make a transition to gain $j = i + 1$. If $2 \leq tx\_count \leq 4$ make a transition as in feedback 0.

Feedback *e* always leads to a state transition upwards.

1 If *bit_error_rate* , together with *gain_state* , requires a gain state increase, make a transition to the gain state determined by these two variables. Else: if *ack_count* < 10 make no transition. Else: if *ack_count* = 10 and *bit_error_rate* , together with *gain_state* -1, indicates that a transition downwards is acceptable, make a transition to state $j = i - 1$. Else: if *ack_count* = 10 but *bit_error_rate* , together with *gain_state* -1, indicates that a transition downwards is **not** acceptable, make a (self) transition to state $j = i$.

These rules assume that bit error statistics are available. If they are not, because there is no FEC (in a modified algorithm) or because too little sample data is available, the decision is simply based on *ack_count* . If *ack_count* 10 make no transition. If *ack_count* = 10 make a transition to $j = i - 1$.

These four if statements cover all possibilities. Feedback 1 is the only feedback type that can result in a gain state decrease. It is also the only one that can result in using bit error statistics, and the only one that can result in a self transition from an intermediate gain state.

Bit Error Computation Bit error computations are only required after observation of feedback of type 1. Variables *bit_count* and *bit_errors* are updated, and the variable r = *bit_error_rate* = *bit_count* *bit_errors* is computed. The current gain state on the link and this bit error rate variable serve as inputs to a function *state_change* (*bit_error_rate* ,*gain_state*). This function returns two values, *transition* and *new_state* . The variable *transition* is binary. It takes the value ON if *gain_state* is not the most favorable gain state, as will be explained below. In this case the *new_state* will contain the most favorable state. *new_state* = *gain_state* implies a self transition. This can happen if more than maximum gain is required to guarantee a robust link. *transition* takes the value OFF if *gain_state* is the most favorable state. This implies no transition. It can also take the value OFF if the function considers that not enough sample errors exist to venture a calculation. This condition is expressed by *bit_errors* < min_errors. We have chosen a value min_errors =3. We expect to fine tune it as we gain more experience with the algorithm. If *transition* = OFF variable *new_state* is set to *gain_state* (in fact its value is irrelevant).

The outputs *transition* and *new_state* are used in the transition rule for feedback of type 1. Here we describe how function *state_change* computes their value. The algorithm uses a piecewise linear function $F(\cdot)$ as an approximation to the function $Q(\sqrt{\cdot})$. Figure 6.5 shows both functions. The approximation is used in the SNR calculations which determine if the current gain state is acceptable. A gain state is acceptable if the estimated channel signal-to-noise ratio, expressed in dBs, plus the SNR gain provided by the choice of FEC rate in that state, is greater than or equal to a parameter θ. θ is the target value of signal-to-noise ratio, expressed in dB. It is defined as the SNR value where the bit error probability $Q\left(\sqrt{\theta}\right)$ leads to a packet error probability P_{pkt} = FAIL = 0.1, for a 3000-bit packet. This definition is motivated by the fairness consideration discussed in Subsection 4.1.

Let r be the current estimate of the channel bit error probability. To find if r is acceptable, it can be translated into a channel SNR value by computing the inverse function $F^{-1}(r)$. This function and its inverse are essentially trigonometric calculations. Appendix A describes them in detail. The SNR value $F^{-1}(r)$ includes the effect of signal power and channel bit rate. It does not include the effect of FEC gain since it is measured at the channel, before decoding. FEC gain is included by defining a distance function $D(r,i)$ as

$$D(r,i) = \theta - \left(F^{-1}(r) + G_i^{FEC}\right) \tag{4.2}$$

This is the SNR difference between the desired SNR and the current SNR, including coding gain. i represents the current gain state and G_i^{FEC} is the FEC gain, in dBs, corresponding to the FEC rate of state i. The algorithm has the relative FEC gain values of Table 3.2 stored in memory for this computation.

To determine the appropriate gain state, the algorithm increases or decreases the state, adding or subtracting *differential* gains from the gain table, until the resulting SNR value is the minimum value still greater than θ which can be obtained from the table. This means that the corresponding state found is the lowest gain state capable of providing acceptable performance according to the gain table. The function *state_change* takes the form:

If *bit_errors* < min_errors set *transition* ← ON, $j \leftarrow i$ and return

Else If $D(r,i) > 0$, attempt increase:
set *transition* ← ON
If $i = M$ set $j \leftarrow i$ and return

Else **repeat** $i \leftarrow i+1$. $D \leftarrow D - \Delta G_{i+1}$ **until** $D < 0$ or $i = M$
$j \leftarrow i$, return

Else $D(r,i) \leq 0$, attempt decrease:
If $i = L$ set *transition* $\leftarrow$ ON, $j \leftarrow i$ and return
Else
$i_0 \leftarrow i$
repeat $i \leftarrow i-1$. $D \leftarrow D + \Delta G_{i+1}$ **until** $D \geq 0$ or $i = L$
If $i = L$ set *transition* $\leftarrow$ ON, $j \leftarrow i$, return
Else ($D \geq 0$) set $j \leftarrow i+1$
if $j = i_0$ set *transition* $\leftarrow$ OFF and return if $j \neq i_0$ set *transition* $\leftarrow$ ON and return

On completion, the function will have found the appropriate gain state and determined if a transition was necessary. Notice that the transition rule for decrease after feedback 1 only calls for a decrease of one step, regardless of the value $j \equiv$ *new_state*, as long as $j < i$. This effectively means that the algorithm prefers to be cautious about decreasing gain even if *state_change* prefers a decrease by more than one gain state. The reasons for this approach are discussed under Section 5.

Summarizing, function *state_change*(r,i) takes as input the estimate of channel bit error rate, r, and the current gain state i, and computes if a transition is necessary to guarantee an acceptable link, that is, $P_{pkt} \leq$ FAIL for a maximum-length packet. It also determines the gain state j which seems to provide sufficient gain for an acceptable link, with the minimum overhead possible. This new state is found by moving up or down the gain table, using the differential gain entries.

Ack Mode The Parameter Selection Algorithm has been explained in terms of adjusting gain based on data packet transmissions and their outcomes. The *ack mode* addresses the problem of adjusting gain on a link with no data traffic. The problem is important because a link (j,i) serves as the feedback channel for link (i,j), which may carry data. If (j,i) carries no data an fails to adapt to noise changes it may fail to adapt an link (i,j) may be condemned to many retransmissions if its acks are swamped by noise.

Even if no data flows through (j,i), traffic flows through this link as long as data traffic flows through (i,j). The traffic is the feedback traffic. This suggests a solution. There is an analogy between data traffic and ack traffic. Ack and nack packets replace data packets. Unwanted retransmissions through (i,j) serve as feedback 0 for link (j,i). New data packets through (i,j) serve as feedback 1 for link (j,i). The analogy is not perfect. Among other things, there is no feedback ϵ. Still, let us redefine the feedback events to suit operation in the ack mode.

Feedback 0, for link (j,i) in the ack mode, refers to reception of an unsolicited retransmission through channel (i,j). It corresponds to reception of a retransmission copy of a data packet already acked or nacked.

Feedback 1 refers to observing no retransmission of an acked, or nacked, data packet. It corresponds to reception of a data packet through link (i,j) different from the last packed from i to have been acked or nacked. The feedback is ambiguous since node j may simply have missed all further retransmissions of a data packet that was acked or nacked.

Feedback ϵ does not exist.

The link state is also redefined in the ack mode. The state is made up of four variables. Two of them, *gain_state* and *data_cycle* , are exactly the same variables being used while in the regular *data*

mode. While *data_cycle* =ON , data traffic updates *gain_state* . While *data_cycle* =OFF , ack traffic is allowed to update *gain_state* . The other two variables are *acks_sent* and *feed1_count* . *acks_sent* , in the ack mode, replaces *tx_count* . *feed1_count* replaces *ack_count* . There are no ack-mode equivalents to *bit_errors* and to *bit_count* .

The ack mode operates as follows. The first transmission of a data packet through link (j, i) resets to zero the ack-mode link-state variables *acks_sent* and *feed1_count* . This transmission also sets *data_cycle* =ON disabling the ack mode as explained earlier. Setting *data_cycle* to OFF , after the data packet has been disposed off, re-enables the ack mode.

Recall that data transmissions through (j, i) use *data_cycle* to ensure that no more gain update takes place after the first acknowledgment to a packet is received. Multiple acknowledgments are possible since they can get delayed. Such delay can lead to several retransmissions before the first ack is returned. The receiving node however must ack all retransmissions. *data_cycle* is just a simple way to guard against the possibility of reacting to stale acks. The ack mode does not need an equivalent of *data_cycle* since there is no ARQ strategy required for acks. In other words, a succession of ack-mode feedback 1 events for acks of a given packet is impossible since only one ack is returned per packet, regardless of how long it takes to hear from the same neighbor again.

The variables *acks_sent* and *feed1_count* in the ack mode are updated just like *tx_count* and *ack_count* in the regular data mode. When feedback arrives, the values are increased if necessary. The same transition rules as for the data mode could be used, except that there would be no feedback e situation and there would be no bit error computations. Instead we introduce two modifications in the rules for feedback 0. The first is a change in the value of *acks_sent* at which noise problems are suspected. If feedback 0 arrives when *acks_sent* =0, no transition is made. If *acks_sent* =1, a transition one step up is made (may result in self transition). For $2 \leq$ *acks_sent* ≤ 4 the same rule as for the data mode is used. The second modification is that, while in ack mode, the ack for the sixth transmission of a data packet causes a transition to the highest gain state regardless of the actual number of acks previously sent for the packet.

The motivation for the first modification above is that acks are protected against contention by the Moment of Silence Algorithm, and against noise by virtue of their small length. Thus, fewer ack retransmissions due to blocking are expected, and we are willing to increase the gain sooner as a function of the number of feedback 0 events observed. The second modification attempts to guarantee that a last data packet transmission will be acknowledged even if previous earlier transmissions of the data packet have not been received. Even though a data packe is disposed of after six transmissions, regardless of feedback, the outcome of the transmission is relevant to the Link Up/Down Protocol in deciding whether to terminate a link or not.

Link Up/Down Protocol The *Link Up/Down Protocol* makes decisions on establishing or terminating a radio link based on information it obtains from the operation of the Parameter Selection Algorithm. The protocol must also interface with the Multi-Class Routing Algorithm.

When a PROP packet is overheard for the first time, the Link Up/Down Protocol attempts to establish a link, provided that the maximum number of neighbors permissible is not being exceeded. Multi-Class routing distinguishes two classes of links, *good* and *bad*. The following is the way in which the Link Up/Down Protocol determines the class of a link.

Because of the Security Architecture proposed by BBN, a link must be authenticated to the satisfaction of the two nodes sharing it before the link is established. The process requires the exchange of several packets, all sent at maximum gain. The protocol computes the fraction of packets exchanged successfully and determines the class based on this computation. Two thresholds, a and b, $0 < a < b < 1$, are used in the decision. If the fraction is less than a the link is not established. An *entry timer* is set which prevents any attempt of establishing the same link again until this timer expires. This prevents erratic topological changes. If the fraction falls between a and b, the link is declared bad, for the purposes of multi-class routing. If the fraction exceeds b the link is declared good. The Parameter Selection Algorithm continues running on the link as explained in previous sections.

The link may be terminated as follows. If six consecutive transmissions at maximum gain go unacknowledged in a good link, the link is declared bad. At this point the fraction of packets successfully transmitted is computed over the next N transmissions. The same parameters a and b are used to determine the link class. A link that remains bad for more than c of these N packet periods is terminated. A link can also be terminated for another reason. If after a number of PROP periods not a single PROP has been received from the corresponding neighbor the link is terminated and the entry timer is set. This allows a link to be terminated even when no traffic is flowing towards this neighbor. It also terminates the link even if traffic flowing to that neighbor is being acknowledged, because maintaining a link to a neighbor for which we obtain no topological information is considered a liability for survivability.

We are waiting to gain experience in realistic simulation scenarios to determine the value of the parameters mentioned. The value of b should be close to one since a packet can attempt six transmissions but we would like to use it only if the number of attempts required is low. The value of a can be more flexible since it will only remain in existence if it manages to return to good status. The value of the entry timer must be chosen according to the maximum rate at which the Routing Algorithm feels comfortable with updating topology. The values c and N relate to the amount of time we are willing to risk having a bad link in use. c should be a very small integer constant while N should be large enough to allow computation of the ratio of successful packets to total packets transmitted. It should probably have the same value as the number of packets used to determine if a link should be established.

Packet Format The Parameter Selection Algorithm requires an exchange of information between the receiving node and the sending node. The use of whacks is one example. The algorithm also requires that bit error information be exchanged. When a data packet is received successfully, that is, when the CRC checks, and if the FEC is in use, the receiving node's acknowledgment must include a count of the number of bits received and the bit error count determined by the sequential decoder. Both numbers are integers under 10000, each of which can be represented within two octets. In fact *bit_count* is unnecessary since the sender can remember it, and *bit_errors* is likely to be much less than 10000, so 1 to 1.5 octets are probably enough for the overhead conscious designer.

The receiving node must also store the *bit_errors* and *bit_count* information in memory. The most recent value of *bit_error_rate*, translated into packet error probability based on the average length of a packet on the link, is used by the Congestion Control Algorithm to factor out retransmissions due to channel noise from the calculation of average number of transmissions per packet to the node. Similarly, the most recent *bit_error_rate* value for an output link is used by the Congestion Control Algorithm to calculate the maximum throughput of that link. Since our design calls for low values of packet error

probabilities. it is likely that such calculations will not lead to significant changes in the operation of SURAP 4.

5. Performance

Section 5 presents an analytical model of the algorithm operation, performance analysis and simulation results.

5.1 Semi-Markov Model

The operation of the algorithm can be represented by a Semi-Markov model, which can be used to compute the steady-state probability distribution of transmission gain, packet error probability, transmission power overhead, and packet duration overhead. Packet error probability refers to noise errors and not to blocking due to contention. Three other important applications of the model are possible. First, the model could include a stationary model of transceiver blocking probability. Second, the model could treat dynamic (state-dependent) CDMA interference. Unfortunately we do not have available a characterization for this interference. Third, the states transitions could be altered to represent alternative algorithms based on similar operating principles. These applications are discussed in detail at the end of this section. To simplify the discussion we will first derive the equations for the Semi-Markov model in a specific channel situation. Then we will derive the equations for the general Semi-Markov model of the algorithm. Neither the specific model nor the general model incorporate algorithm transitions resulting from bit error calculations but we will propose a simple way to include these transitions.

The *specific* Semi-Markov model will assume that there is a stationary signal-to-noise ratio in the channel, no contention for the transceiver, and no dynamic CDMA interference. The last two assumptions are unnecessary for the general model. We assume that there is an instantaneous and perfect feedback channel. A transmission at gain state i results in three possible feedback types: success, whack, and no reply. In the feedback notation of Section 4.2 they are denoted 1, ϵ, and 0, respectively. The gain state and the signal-to-noise ratio in the channel define the feedback probabilities, $p_1(i), p_\epsilon(i), p_0(i)$. They can be computed as explained in Section 4.1. In the model being discussed, $p_0(i)$ refers to a packet so corrupted by noise that the receiving node is unable to return a whack to the packet sender. In general, this probability could also represent transceiver blocking probability and CDMA interference error probability.

The key factor that allows a relatively simple model for the process is to treat a sequence of acknowledged transmissions as a random wait in a Markov state representing the gain index instead of treating the sequence as a walk through individual Markov states. This means that the wait for ten successful transmissions to decrease gain need not be represented as ten states in the model. The only system states that need to be modeled are the gain state and the number of transmission attempts elapsed for the packet being forwarded.

Steady-State Probabilities A Semi-Markov process is a Markov process which spends a random amount of time at each state before the next transition. The time spent at a state is the *holding time*. We will define a two-dimensional Discrete-Time Semi-Markov process to model the algorithm. The two dimensions correspond to the gain state in the link and to the number of transmissions attempted by a packet that enters this gain state. However, one auxiliary state will be needed per gain state. The reader should not confuse states in the Markov chain with gain states. The gain states are only one dimension in the two-dimensional Markov state space. The discrete holding times are the number of consecutive packet transmissions and retransmissions once a Markov state is entered. The transition instants are related to the instants when a packet is acknowledged or the link gain state is changed[1]. All transition probabilities in the Markov chain and all holding time distributions can be computed from the probabilities $p_1(i), p_e(i), p_0(i)$. In the general case, with 7 gain states and up to 6 transmission attempts per packet, 38 Markov states will be needed. The transition probability matrix representing the Markov chain is sparse. Standard linear algebra techniques can be used to find the steady-state probabilities $\pi_{i,j}$ for the Markov chain. Our specific example simplifies to 11 Markov states. The expected values $\bar{t}_{[i,j]}$ of the state holding times are also straightforward to compute. From these, the probability distribution for the steady state gain state i in use is

$$g(i) = \frac{\sum_j \pi_{i,j} \bar{t}_{[i,j]}}{\sum_l \sum_j \pi_{l,j} \bar{t}_{[l,j]}} \tag{5.1}$$

The probability of unsuccessful transmission (no acknowledgement), $f(i)$, is only dependent on the state i. Hence the packet error probability for the model, P_{pkt}, is found from

$$P_{\text{pkt}} = \sum_i g(i) f(i) \tag{5.2}$$

P_{pkt} depends on the SNR through the feedback probabilities $p_1(i), p_e(i), p_0(i)$. We can also find the transmission power overhead and packet duration overhead by substituting the overhead value per gain state for the probabilities $f(i)$ in equation (5.2).

State Space Figure 6.6 shows a detailed diagram of a two-dimensional Markov model for the algorithm's operation in the illustrative example. The system state is denoted by $S(i, j)$. State variable i denotes the gain state. Gain state I corresponds to a state state near which the algorithm is expected to operate in steady state so that the average packet error probability is kept below the parameter FAIL introduced in Section 4.2. Only two gain states above and below I are included in the model, because excursions outside these states are rare. It is no harder to incorporate more gain states. State variable j, for $j \geq 0$ in $S(i, j)$, denotes the number of previous transmissions attempts for the packet that enters gain state i. We denote by B the number of successful transmissions that lead to a decrease in gain state. $B = 10$ in our algorithm. The auxiliary state variable $j = -1$ allows the model to remember when one successful transmission has already taken place at the current gain state.

[1]Self-transitions of gain state, as described in section 4.2, are also transitions in the Markov state space.

Transitions We now explain the state transitions in the diagram of Figure 6.6. The assumptions that we adopt for simplicity of exposition are unnecessary in the general model. The first simplifying assumption is that in gain states $i = I, I+1$, in the absence of contention, a transmission is either successful or results in a whack. This is, $p_0(i) \simeq 0$, $i = I, I+1$. Similarly, we assume that all transmissions at gain state $i = I+2$ result in an ack while the probabilities of whack or acks in gain state $i = I-2$ are zero. That is, $p_1(I+2) \simeq 1, p_0(I-2) \simeq 1$. For concreteness, this model is applicable to a channel which experiences a signal-to-noise ratio of about -19 dB at the minimum gain state 0, when the average packet length is 150 bits and packets have a 16-bit sender-identity field. I represents gain state 4, in the gain states 0 through 6. This specific example was chosen because it illustrates the main points that need discussion, even though these network parameter values are perfectly feasible.

In Figure 6.6 the state transition labels stand for the event causing the transition. Label 1 stands for ack reception. Label B and $B-1$ stand for B and $B-1$ consecutive data packets successfully forwarded at the same gain state. Label ϵ stands for whack reception on the first transmission attempt of a packet. Label $0.\epsilon$ stands for a succession of no reply and whack feedback on the first two transmission attempts of a packet. Similar notation is implied in labels $0.0.0$ and $0.0.\overline{1}$. The symbol $\overline{1}$ refers to reception of either a whack or no reply.

Let us describe the transitions in detail starting with state $S(I-2.0)$

$S(I-2,0)$ Since $p_0(I-2) \simeq 1$, the system leaves this Markov state after three transmission attempts. The gain state increases to $i = I$, in accordance with equation (4.1). The transition is to state $S(I.3)$ since the packet has been through three transmission attempts already.

$S(I-1.0)$ In this Markov state, the system attempts up to three transmissions per data packet. If any attempt results in an ack, a new data packet is tried up to a maximum of B data packets. The system decreases the gain state only if it can forward B consecutive data packets at the current gain state. In this case the Markov state changes to $S(I-2.0)$. Three more state transitions are possible and all of them lead to an increase of gain state. First, a whack for the first transmission attempt of a data packet leads to an increase of one in the gain state. The Markov state should change to $S(I.1)$. Instead we chose to draw the transition into $S(I.0)$. This alteration is not necessary but simplifies the Markov state space. The reason for the j dimension in the model is to be able to model gain state increases after three or more unsuccessful attempts. But since $p_0(i) \simeq 0$ for $i = I$ and above in the example under discussion, the system would simply increase the gain state by one after each whack is received, or attempt to send a new data packet if an ack is received. Either situation is properly represented in the simplified model. This simplification may appear inelegant, but it illustrates an important way to reduce the Markov state space. A second transition corresponds to a feedback sequence $0.\epsilon$. The system increases the gain state by one, and enters Markov state $S(I.2)$. The third transition remaining corresponds to a feedback sequence $0.0.\overline{1}$. The system increases the gain state to $i = I$, according to equation (4.1), and enters Markov state $S(I.3)$. These last two transitions discussed could have been simplified as we simplified the transition due to a whack on first packet transmission attempts. We choose not to simplify them so that we can explain the basis for the model.

$S(I.0)$ Transitions from this Markov state are a simpler version of the transitions from $S(I-1.0)$, since $p_0(I) \simeq 0$, so this time transitions can only result in whacks or acknowledgments. Again we have

modified the transition corresponding to a whack, as for Markov state $S(I-1,0)$ above.

$S(I,2)$, $S(I,3)$ A whack in these states results in a transition to $i = I+1$. An ack in these two states results in a transition to $S(I,-1)$.

$S(I,-1)$ The system decreases the gain state, as shown, after only $B-1$ consecutive successes, since the previous packet transmission took place at the same gain state and was successful. A whack transition leads to an increase of gain state.

$S(I+1,j)$ Transitions out of these states, $j = -1\ldots4$, are similar to those from $S(I,j)$. States $S(I+1,1)$ and $S(I+1,2)$ are unnecessary because of the earlier simplifications of whack transitions out of Markov states $S(I-1,0)$ and $S(I,0)$.

$S(I+2,0)$ All transmissions at this state result in acknowledgment since $p_1(I+2) \simeq 1$, so the Markov state transition is as shown.

Transition Probabilities Let $P[m,n\,|\,i,j]$ denote the transition probability from Markov state $S(i,j)$ to Markov state $S(m,n)$. The expressions for the transition probabilities appear in Table 5.2. The explanation that follows groups them by the state occupied before transition and explains how the expression for each of the transition probabilities that applies is found.

$S(I-2,0)$ $P[I,3\,|\,I-2,0] = 1$ since we assume $p_0(I-2) = 1$.

$S(I-1,0)$ Define $P_{\mathrm{FWD}}(i)$ as the probability that an ack is received within the first three transmission attempts of a packet, given that the packet is sent at gain state i.

$$P_{\mathrm{FWD}}(i) = p_1(i)(1 + p_0(i) + p_0(i)^2) \tag{5.3}$$

In Table 5.2, $P[I-2,0\,|\,I-1,0]$ is the probability that B packets are successfully forwarded within three transmission attempts each. For the other transitions define the three conditional probabilities $P_2(i)$, $P_{0,2}(i)$, $P_{00\bar{1}}(i)$. These correspond to the three possible feedback sequences in the first three transmission attempts of a packet sent at gain state i, given that none of the three attempts results in an ack.

$$P_2(i) = \frac{p_2(i)}{1 - P_{\mathrm{FWD}}(i)} \tag{5.4}$$

$$P_{0,2}(i) = \frac{p_0(i)p_2(i)}{1 - P_{\mathrm{FWD}}(i)} \tag{5.5}$$

$$P_{00\bar{1}}(i) = \frac{p_0(i)p_0(i)(1 - p_1(i))}{1 - P_{\mathrm{FWD}}(i)} \tag{5.6}$$

The probability that the transition out of the current Markov state consists of any of these three events is $1 - P_{\mathrm{FWD}}(i)^B$. The product of this probability times each of the three conditional probabilities above gives the entries for the transition probabilities $P[I,0\,|\,I-1,0]$, $P[I,2\,|\,I-1,0]$, $P[I,3\,|\,I-1,0]$ in Table 5.2.

Table 5.1: Auxiliary Equations for Transition Probabilities.

$$P_{FWD}(i) = p_1(i)(1 + p_0(i) + p_0(i)^2)$$

$$P_c(i) = \frac{p_c(i)}{1 - P_{FWD}(i)}$$

$$P_{0,e}(i) = \frac{p_0(i)p_c(i)}{1 - P_{FWD}(i)}$$

$$P_{0,0\,\bar{1}}(i) = \frac{p_0(i)p_c(i)(1 - p_1(i))}{1 - P_{FWD}(i)}$$

$S(I, 0)$ Probability $P[I-1.0\ I.0]$ is analogous to $P[I-2.0\ I-1.0]$ except that $ForwardProb(I) = p_1(I)$ since only acks or whacks are possible. $P[I+1.0\ I.0]$ is also analogous to $P[I.0\ I-1.0]$ except that the conditional probability $P_c(i)$ equals 1 and that $P_{0,e}(i)$, $P_{0,0\,\bar{1}}(i)$ equal 0.

$S(I, 2)$, $S(I.3)$ In both cases the transition probabilities are given by the probability of receiving a whack or by the probability of receiving an ack.

$S(I, -1)$ $P[I+1.0\ I.-1]$ is analogous to $P[I+1.0\ I.0]$. $P[I-1,0\ I.-1]$ is analogous to $P[I-1.0\ I.0]$ except that only $B-1$ consecutive successes are required to decrease the gain state.

$S(I+1, j)$ The transition probabilities out of states with $i = I+1$ are analogous to the probabilities for states with $i = I$ described above.

$S(I+2, 0)$ $P[I+1,0\ I+2.0] = 1$ since $P_{FWD}(i) = p_1(i) = 1$.

Steady-State Solution Once the transition probabilities have been assigned, the transition matrix **P** for the embedded Markov chain can be computed. The steady-state probabilities $\pi_{i,j}$ for the chain are the solution of the simultaneous set of linear equations

$$\pi = \pi \mathbf{P} \tag{5.7}$$

$$\sum_{i,j} \pi_{i,j} = 1 \tag{5.8}$$

where π is the row vector listing the steady-state probabilities for each of the states in the chain. Matrix inversion solutions of these equation can be obtained with complexity of order n^2 or less.

Holding Times Table 5.3 summarizes the expected value for all holding times in the chain of Figure 6.6.

The holding times for several states in Figure 6.6 are deterministic. In particular, $\bar{h}[i,j] = 1$ for any i when $j > 0$, $\bar{h}[I\ -2\ 0] = 3$, and $\bar{h}[I\ -2\ 0] = 10$.

Table 5.2: Transition ProbabilitiesSpecific Model

from $S(I-2,0)$	$P[I,3\ I-2.0]$	1
from $S(I-1.0)$	$P[I-2,0\ I-1.0]$	$P_{\mathrm{FWD}}(I-1)^B$
	$P[I.0\|I-1.0]$	$(1-P_{\mathrm{FWD}}(I-1)^B)P_e(I-1)$
	$P[I,2\ I-1.0]$	$(1-P_{\mathrm{FWD}}(I-1)^B)P_{0,e}(I-1)$
	$P[I,3\|I-1,0]$	$(1-P_{\mathrm{FWD}}(I-1)^B)P_{0,0,\bar{1}}(I-1)$
from $S(I.j)$	$P[I-1.0\ I,0]$	$p_1(I)^B$
	$P[I+1.0\ I.0]$	$1-p_1(I)^B$
	$P[I-1,0\ I,-1]$	$p_1(I)^{B-1}$
	$P[I+1.0\ I.-1]$	$1-p_1(I)^{B-1}$
	$P[I.-1\ I.2]$	$p_1(I)$
	$P[I+1.3\ I.2]$	$p_e(I)$
	$P[I,-1\|I.3]$	$p_1(I)$
	$P[I+1.4\ I.3]$	$p_e(I)$
from $S(I+1,j)$	$P[I,0\|I+1,0]$	$p_1(I+1)^B$
	$P[I+2.0\ I+1,0]$	$1-p_1(I+1)^B$
	$P[I.0\|I+1.-1]$	$p_1(I+1)^{B-1}$
	$P[I+2.0\ I+1,-1]$	$1-p_1(I+1)^{B-1}$
	$P[I+1.-1\ I+1.3]$	$p_1(I+1)$
	$P[I+2.0\ I+1.3]$	$p_e(I+1)$
	$P[I+1.\ \ 1\ I+1.4]$	$p_1(I+1)$
	$P[I+2.0\ I+\ .4]$	$p\ (I+1)$
from $S(I+2.0)$	$P[I+1.0\ I+2,0]$	1

For the remaining states it will be useful to derive equation (5.13) below. Consider a Bernoulli process where at each Bernoulli trial k the discrete random variable R_k has distribution $F_R^{(1)}[r]$ with probability p and has distribution $F_R^{(2)}[r]$ with probability $1 - p$. These two distributions are independent of the trial state k and so is the probability p. The process stops the first time that R_k is distributed according to $F_R^{(2)}[r]$ or when N_{max} Bernoulli trials have occurred. We define random variable L as

$$L = \sum_{k=1}^{N} R_k \tag{5.9}$$

where N is the trial at which the process stops. The result we are after is the expected value $E[L]$. Since the sequence R_k consists of independent, identically distributed random variables

$$E[L] = E[N]E[R] \tag{5.10}$$

The factor $E[R]$ is simply the expected value of R_k, which is independent of k. To find $E[N]$ we note that N is a discrete, non-negative random variable. If $F_N[n]$ is its distribution function and $F_N^c[n] = 1 - F_N[n]$ is the complementary distribution function.

$$E[N] = \sum_{n=0}^{N_{max}} F_N^c[n] \tag{5.11}$$

By looking at the event tree shown in Figure 6.7 for the Bernoulli process, we find that for the values $n = 0, 1, 2, \cdots, N_{max} - 1, N_{max}$ we obtain $F_N^c[n] = 1, p, p^2, \cdots, p^{N_{max}-1}, 0$. Hence

$$E[N] = \sum_{n=0}^{N_{max}-1} p^n = \frac{1 - p^{N_{max}}}{1 - p} \tag{5.12}$$

From equation (5.10) we arrive at

$$E[L] = \left(\frac{1 - p^{N_{max}}}{1 - p} \right) E[R] \tag{5.13}$$

We can now find the expected values for the remaining non-deterministic holding times. In state $S(I - 1, 0)$ the maximum number of trials is B. The trials correspond to attempting to forward a data packet, and the probability p that a new trial will be attempted is $P_{FWD}(I - 1)$. The random variable R is now the number of packet attempts until receiving an acknowledgement, a whack, or making three attempts to send a packet. This is in fact the same situation we had for calculating $E[N]$ in equation (5.12), but replacing 3 for N_{max} and $p_0(I - 1)$ for p. The result is $(1 - p_0(I - 1)^3) \cdot (1 - p_0(I - 1))$. From equation (5.13) we obtain

$$\bar{t}_{[I-1,0]} = \left(\frac{1 - P_{FWD}(I - 1)^B}{1 - P_{FWD}(I - 1)} \right) \frac{(1 - p_0(I - 1)^3)}{1 - p_0(I - 1)} \tag{5.14}$$

For the remaining non-deterministic holding times in our model the number of attempts per packet is one so the term $E[R]$ in equation (5.13) equals one. For states $S(i, i - 1), i = I, I + 1$, the maximum number of trials is $B - 1$, and for the other two states $S(i, 0), i = I, I + 1$, the maximum number of trials is B.

Table 5.3: Expected Holding Times for Specific Model

State	*Expected Holding Time*
$S(I-2,0)$	3
$S(I-1,0)$	$\left(\frac{1-P_{FWD}(I-1)^{E}}{1-P_{FWD}(I-1)}\right)\frac{(1-P_{FWD}(I-1)^{3})}{1-P_{FWD}(I-1)}$
$S(I,0)$	$\left(\frac{1-P_{FWD}(I)^{E}}{1-P_{FWD}(I)}\right)$
$S(I,-1)$	$\left(\frac{1-P_{FWD}(I)^{E-1}}{1-P_{FWD}(I)}\right)$
$S(I,2)$	1
$S(I,3)$	1
$S(I+1,0)$	$\left(\frac{1-P_{FWD}(I+1)^{E}}{1-P_{FWD}(I+1)}\right)$
$S(I+1,-1)$	$\left(\frac{1-P_{FWD}(I+1)^{E-1}}{1-P_{FWD}(I+1)}\right)$
$S(I+1,3)$	1
$S(I+1,4)$	1
$S(I+2,0)$	10

Table 5.4: Evaluation Of Feedback Probabilities.

Gain State i	$p_1(i)$	$p_e(i)$	$p_0(i)$
$I-2=2$	~ 0	0.063	0.937
$I-1=3$	0.408	0.501	0.091
$I=4$	0.589	0.356	0.055
$I+1=5$	0.987	0.011	0.001
$I+2=6$	~ 1	~ 0	~ 0

Comparison to Simulation We checked the validity of the Semi-Markov model in the example discussed, by running a simulation of a version of the Parameter Selection Algorithm that ignores bit error calculations. The simulation confirmed that the model gives accurate results for this test. The simulator used is described in Appendix B.

The situation simulated consisted of 500 data packets to be forwarded on a contention-less link. The channel noise was stationary, resulting in a channel signal-to-noise ratio of -23 dB at gain state 0. Once the FEC gain for FEC rate 7/8 is included, the signal-to-noise ratio perceived by the data bits is (nominally) -19 dB. We simulated constant length packets of 150 bits, with sender-id fields of 16 bits.

To check the validity of the assumptions on the relative values of the feedback probabilities $p_1(i), p_e(i)$ and $p_0(i)$ we evaluated the probabilities for all five gain states represented in the model. These values appear in Table 5.4.

The notation ~ 0 represents a value less than 10^{-10}. The notation ~ 1 represents a value between $1-10^{-10}$ and 1. Feedback 0 is the dominant form of feedback at gain state $I-2$. All three feedback outcomes were included in the model for gain state $I-1$. The probability for feedback 0 in this gain state is not much higher than that of feedback e at gain state $I-2$, or feedback 0 at gain state I. However we chose to include feedback 0 in the model for gain $I-1$ because it allowed us to describe the general case of ternary feedback. For gain state $I+1$ only feedbacks e and 1 are considered relevant. For gain state $I+2$ only feedback 1 is relevant.

The simplification of feedback outcomes for each state and the simplifications on transition possibilities discussed earlier are purely a matter of convenience in the specific model being considered. The general Semi-Markov model does not require any of these simplifications.

The outcome of the computer simulation showed the following number of transmissions at gain states 0 through 6, in that order: 3,0,1,0,84, 401, 64. The total number of transmissions is 553. This number exceeds 500 (the total number of data packets forwarded) because of packet retransmissions. The transmissions at gain states 0 and 2 occur because the algorithm begins the simulation at gain state 0. These transmissions are the transient behavior of the algorithm. They have a minimal effect on the

computation of the average gain state distribution. The average distribution for gain states 0 through 6, is expressed to two decimal places in the following percentages: 0%, 0%, 0%, 0%, 15%, 73%, 12%. The following percentages represent the distribution computed from the Semi-Markov model 0%, 0%, 0%, 0%, 18%, 73%, 9%. The average packet error probability computed from the simulation is $1 - 500,553 = 0.096$ while this probability computed from the model is 0.083. The model gives accurate results in the probability of error and, more importantly, in the gain state distribution.

Generalized Semi-Markov Model We can now generalize the model. Three different sets of Markov states must be considered. Two sets of *boundary gain states* corresponding to the maximum and minimum gain state, $i = M$ and $i = L$, and *intermediate gain states* corresponding to gain states between M and L. These states and their transitions appear in figures 6.8 and 6.9. The explanation follows.

A set of six Markov states are required for each intermediate gain state. These are $S(i,-1)$, $S(i,0)$, $S(i,1)\cdots S(i,4)$. State $S(i,5)$ is not required since if five transmissions attempts for a packet have occurred, the next Markov state should be $S(M,5)$, where M is the highest gain state. For $i = M$, the set of Markov states required are $S(i,-1), S(i,0), S(i,1)\cdots S(i,5)$. For $i = L$, only one Markov state is required: $S(L,0)$. In this last case the system remains in the state until it fails to forward a packet within three transmission attempts, and the gain is increased, or until B packets in a row are forwarded within three transmission attempts each, resulting in a self-transition to the same Markov state.

The transition probabilities appear summarized in Tables 5.5 and 5.6. Expected values of holding times appear in Table 5.7. We start with transitions from the Markov states corresponding to the intermediate gain states. (See Figure 6.8.) We denote by i_3, i_4, i_5 the gain values computed after an unsuccessful third, fourth and fifth transmission attempt respectively (equation (4.1)). Transitions from Markov state $S(i,0)$ are similar to the transitions from state $S(I-1,0)$ in the model of Figure 6.6. One difference exists. There is no transition to state $S(i+1,0)$, corresponding to the transition to $S(I,0)$ in the earlier model. Instead there is a transition to $S(i+1,1)$. In Figure 6.6 this transition was altered to simplify the model. The transition probabilities remain the same as before. The expected values of the holding times (Table 5.7) are similar to the expressions for the Semi-Markov model in the illustrative example.

Transitions from $S(i,-1)$ are similar to transitions form $S(i,0)$. In the earlier example, transitions out of $S(i,-1)$ were simpler to treat since there was no need to model the possibility of 0 feedback.

Transitions from state $S(i,1)$ have no parallel in the earlier model, yet they turn out to be a simplified version of the transitions from state $S(i,0)$. There are three possibilities: an acknowledgment in the second or third transmission attempt, a whack on the second transmission attempt, or no reply in the second attempt followed by no ack in the third attempt. Transitions are to Markov states $S(i,-1), S(i+1,2), S(i_3,3)$, respectively. At most one data packet will be forwarded in this Markov state before a transition. The expected value of the holding time can be found in a fashion analogous to finding the expected value of the random variable R, for equation (5.14) with two attempts at most, instead of three. The result is $(1-p_0(i)^2)/(1-p_0(i)) = 1 - p_0(i)$, shown in Table 5.7.

Transitions from states $S(i,2), S(i,3), S(i,4)$ are given by the probability of receiving or not receiving an ack after the transmission. Each of these states has a holding time equal to one transmission.

For boundary states (see Figure 6.9) corresponding to $i = M$, transitions to Markov state $S(M,0)$ occurr if any packet fails to be forwarded within three transmission attempts. These transitions correspond to the gain state self-transitions explained in Section 4.2. For the boundary state $S(L,0)$, where L denotes

the lowest gain state, a self-transition occurs each time an attempt is made to further reduce the gain state. The expressions for the expected values of the holding times for all boundary Markov states are straightforward applications of earlier expected holding time calculations.

The general model can represent transceiver contention because it includes the possibility of 0 feedback. It can also include state-dependent CDMA interference. CDMA interference may cause gain increases in one link to trigger gain increases in adjacent links. This situation can be modeled by a state-dependent signal-to-noise ratio representing the CDMA interference. It would be necessary to obtain first the characterization of this signal-to-noise ratio as a function of gain state. Alternative algorithms can be modeled by altering the Markov state transitions to represent the rules desired.

Bit Error Calculations The Semi-Markov model presented does not include transitions resulting from bit error calculations. In Section 5.2, we illustrate that bit error calculations can have a significant impact on the algorithm performance, especially with small packet traffic.

Because bit error statistics can be accumulated over several packets, an accurate description of these statistics would require Markov states that keep track of consecutive packet transmissions at the same gain state. However, avoiding these states was the factor that allowed the relative simplicity of the Semi-Markov model of the algorithm. We propose an approximate model for bit error calculations, which does not require the addition of any Markov states to the Semi-Markov model discussed.

To incorporate the bit error calculations we must model two situations. The first is the possibility that bit error calculations prevent a decrease in gain state. This may occur after successfully forwarding B packets at the same gain. This possibility can be modeled by modifying the probability of transition B (see Figure 6.8) out of states $S(i,0)$. The probability of this transition is modified by a factor equal to the probability that the bit error calculations permit a gain state decrease. This factor is computed as the probability that the signal-to-noise ratio calculated after B packet forwarding intervals is uncacceptable for maintaining a fair and robust link.

The second situation that must be modeled is the possibility that bit errror calculations trigger an increase in gain state. This possibility presents a complication for the Semi-Markov model, but we can model the situation approximately by assuming that only the bits received within one packet forwarding interval are used in the bit error calculation, given that the calculations result in a gain state increase. This is a simplifying approximation whose validity would have to be tested against simulation results. Computing the probability of a gain state increase based on bit error calculations is analogous to computing the probability that bit error calculations will permit a decrease in gain state. The probability of gain state increase must be computed for each possible feedback sequence ending in an ack within the first three transmission attempts at Markov states $S(i,0)$ and $S(i,-1)$. These sequences are: 1; 0,1; 0,0,1. The probability must also be computed for feedback 1 events at Markov states $S(i,j)$, $j > 0$. Once computed, this probability must be added to the transition probabilities corresponding to gain state increases out of these Markov states. When bit error calculations may lead to gain state increases of more than one state, it is necessary to add extra upward transitions out of the appropriate Markov states.

Modeling bit error calculations is necessary for an accurate characterization of the algorithm. Nevertheless, the Semi-Markov model presented remains an important tool analyzing the algorithm's behavior.

Table 5.5: Transition Probabilities. General Model. Intermediate States.

Intermediate States : *Gain State* i		
from $S(i,0)$	$P[i-1,0\ i,0]$	$P_{\mathrm{FWD}}(i)^B$
	$P[i+1,1\ i,0]$	$(1-P_{\mathrm{FWD}}(i)^B)P_{-}(i)$
	$P[i+1,2\ i,0]$	$(1-P_{\mathrm{FWD}}(i)^B)P_{0,e}(i)$
	$P[i_3,3\ i,0]$	$(1-P_{\mathrm{FWD}}(i)^B)P_{0,0,\bar{1}}(i)$
from $S(i,-1)$	$P[i-1,0\ i,-1]$	$P_{\mathrm{FWD}}(i)^{B-1}$
	$P[i+1,1\ i,-1]$	$(1-P_{\mathrm{FWD}}(i)^{B-1})P_{e}(i)$
	$P[i+1,2\ i,-1]$	$(1-P_{\mathrm{FWD}}(i)^{B-1})P_{0,-}(i)$
	$P[i_3,3\ i,-1]$	$(1-P_{\mathrm{FWD}}(i)^{B-1})P_{0,0,\bar{1}}(i)$
from $S(i,1)$	$P[i,-1\ i,1]$	$p_1(i)(1+p_0(i))$
	$P[i+1,2\ i,1]$	$p_e(i)$
	$P[i_3,3\ i,1]$	$p_0(i)(1-p_1(i))$
from $S(i,2)$	$P[i,-1\ i,2]$	$p_1(i)$
	$P[i_3,3\ i,2]$	$1-p_1(i)$
from $S(i,3)$	$P[i,-1\ i,3]$	$p_1(i)$
	$P[i_4,4\ i,3]$	$1-p_1(i)$
from $S(i,4)$	$P[i,-1\ i,4]$	$p_1(i)$
	$P[M,5\ i,4]$	$1-p_1(i)$

Table 5.6: Transition Probabilities. General Model. Boundary States.

Boundary States : Minimum Gain L		
from $S(L,0)$	$P[L,0 \mid L,0]$	$P_{\text{FWD}}(L)^B$
	$P[L+1,1 \mid L,0]$	$(1 - P_{\text{FWD}}(L)^B)P_2(L)$
	$P[L+1,2 \mid L,0]$	$(1 - P_{\text{FWD}}(L)^B)P_{0,1}(L)$
	$P[L_3,3 \mid L,0]$	$(1 - P_{\text{FWD}}(L)^B)P_{0,0,1}(L)$
Boundary States : Maximum Gain M		
from $S(M,0)$	$P[M-1,0 \mid M,0]$	$P_{\text{FWD}}(M)^B$
	$P[M,0 \mid M,0]$	$(1 - P_{\text{FWD}}(M)^B)$
from $S(M,-1)$	$P[M-1,0 \mid M,-1]$	$P_{\text{FWD}}(M)^{B-1}$
	$P[M,0 \mid M,-1]$	$(1 - P_{\text{FWD}}(M)^{B-1})$
from $S(M,1)$	$P[M,-1 \mid M,1]$	$p_1(M)(1+p_0(M))$
	$P[M,0 \mid M,1]$	$p_2(M) + p_0(M)(1 - p_1(M))$
from $S(M,j)$ $j > 1$	$P[M,-1 \mid M,j]$	$p_1(M)$
	$P[M,0 \mid M,j]$	$1 - p_1(M)$

Table 5.7: Expected Holding Times for the General Model.

State	*Expected Holding Time*
$S(i,0)$	$\frac{1-P_{FWD(i)}^{L}}{1-P_{FWD(i)}} \left(\frac{(1-p_0(i)^3)}{1-p_0(i)}\right.$
$S(i,-1)$	$\left(\frac{1-P_{FWD(i)}^{L-1}}{1-P_{FWD(i)}}\right)\frac{(1-p_0(i)^3)}{1-p_0(i)}$
$S(i,1)$	$1+p_0(i)$
$S(i,2)$	1
$S(i,3)$	1
$S(i,4)$	1
$S(M,5)$	1
$S(L,0)$	$\left(\frac{1-P_{FWD(L)}^{L}}{1-P_{FWD(L)}}\right)\frac{(1-p_0(L)^3)}{1-p_0(L)}$

Table 5.8: Packet Error Rates vs. SNR

SNR (dB)	*Pkt Error Prob*
0	0.081
-5	0.016
-10	0.099
-15	0.009
-20	0.082
-25	0.084
-30	1.0

5.2 Packet Error Probability

Error probability The Semi-Markov model is the most powerful tool we have for analyzing steady-state algorithm performance. Unfortunately we do not yet have a general characterization of the algorithm based on the model at the time of writing this report. However, we already had computer simulation results for several channel SNR values spanning the dynamic range of the algorithm showing that the algorithm maintained an acceptable packet error probability for these values.

The Parameter Selection Algorithm has a rule by which it ignores feedback 0 during the first two transmission attempts of a packet. This rule was created to accomodate moderate levels of contention. However, after creating the rule we were concerned that it would increase the average packet error probability to an unacceptable level. For example, if gain state 3 provides acceptable operation, the rule may force the value of packet error probability to equal 0.13 in the following way. After successfully forwarding 10 consecutive packets at gain state 3, the algorithm decreases the gain by 8 dB's to gain state 2. If the next three transmission attempts of the new packet result in a feedback sequence of 0, 0, 0, the gain state will increase to 4, according to equation (4.1). The algorithm will typically experience 20 consecutive successful transmissions and be ready to decrease the gain state to 2 again. This cycle implies $3/23 = 0.13 > 0.1$ packet error rate. The value 0.13, however, is close to the maximum error rate desired. Whacks will tend to reduce this error probability further. The use of bit error statistics also reduces this error probability.

There are many situations similar to the one described above. The results of simulations suggest that the algorithm is able to maintain the packet error probability within the desired range. If in the course of obtaining a more complete evaluation of packet error probability versus channel SNR we find that the error probability becomes unacceptable, we can replace a larger integer value for the number of consecutive successes leading to a gain state decrease. The current value of this number is 10. Increasing this value, however, makes the algorithm less responsive.

Table 5.8 presents simulation results on packet error rate for the Parameter Selection Algorithm. It shows the packet error rate achieved for test values of the SNR experienced by gain state 0. These SNR values are equal to the channel SNR, expressed in dB, plus the coding gain of 4 dB assigned to the 7/8 FEC coding rate for the discussions in this report. The packet error probabilities were obtained using the simulator described in Appendix B.

Table 5.9: Performance without BER Calculations

SNR	*Gain State*							*Pkt Error*
(dB)	0	1	2	3	4	5	6	*Prob*
-14	3	0	71	420	80	0	0	0.129
-15	3	0	46	257	228	30	0	0.110
-16	3	0	10	100	232	200	0	0.083
-17	3	0	1	2	140	400	0	0.084
-18	3	0	1	0	56	421	70	0.093
-19	3	0	1	0	17	285	238	0.081
-20	3	0	1	0	4	165	375	0.088

The simulation consisted of 500 data packets to be forwarded on a contention-less link. The channel noise was stationary and the signal-to-noise ratios tested appear in the table We simulated constant length packets of 1000 bits, with sender-id fields of 16 bits. The table is evidence of the ability of the Parameter Selection algorithm to maintain a robust link over a range of more than 25 dB of stationary noise power.

Tables 5.9 and 5.10 are presented to illustrated the advantage of using bit error calculations. Table 5.9 shows simulation results for a version of the Parameter Selection Algorithm that does not use bit error calculations. The parameters for these simulations are as for Table 5.8. It shows the distribution of packet transmissions over the 7 possible gain states and the packet error rate achieved for selected SNR values. The SNR values refer to the signal-to-noise ratio experienced by gain state 0. The average gain state in use lies between gain state 3 and gain state 6 for the SNR range shown. It is in this range that bit error calculations are expected to be most useful since the difference in gains between gain states is not as large as for gain state changes between transmitter power levels.

The results should now be compared to the results obtained by the Parameter Selection Algorithm under the same tests. These results appear in Table 5.10. Two important differences are worth noticing. The first is the significant decrease in packet error rate. The values of packet error rate corresponding to SNR values from -14dB though -17dB are consistent with a 0.1 packet error rate for 3000-bit packets. For lower SNR values the bit error rate calculations are not as effective. The second difference is the smaller variance of the gain state required to maintain the desired packet error rate. This shows that the bit error rate calculations operated as good estimators of the required signal-to-noise ratio gain required to maintain a given packet error rate.

Fairness The algorithm attempts to provide an acceptable packet error probability for maximum length packets when bit error statistics are available. A trade-off exits between fairness and channel overhead. By virtue of their length, small packets experience lower packet error probability than large packets when bit errors are independent identically distributed bit-to-bit. Thus, small packets can be transmitted at a lower average gain state than larger packets can, for a given packet error value. When bit error statistics are being used, fairness considerations will result in a higher average gain state than is necessary for small packets, and the channel overhead will be larger than required. A compromise between fairness

Table 5.10: Performance with BER Calculations

SNR	*Gain State*							*Pkt Error*
(dB)	0	1	2	3	4	5	6	*Prob*
-14	3	0	1	0	53	448	0	0.010
-15	3	0	1	0	53	438	0	0.010
-16	3	0	1	0	4	487	10	0.010
-17	3	0	1	0	4	478	20	0.012
-18	3	0	1	0	1	454	50	0.018
-19	3	0	1	0	1	288	235	0.053
-20	3	0	1	0	1	110	425	0.082

and channel overhead is certainly possible. Simulation experience can help to decide which factor is most critical.

Whack performance Whacks are used to help discriminate between contention and channel noise. As an illustration consider a channel where the signal-to-noise ratio SNRperceived by gain state 0 varies from 9 dBs down to 1 dB. This SNRvalue is a direct measure of the channel noise level. Assume medium-size packets of 1500 bits are being transmitted and that the sender identity field in the packets is 16 bits long. 9 dBs imply a packet error probability just below 10% for gain state 0. 1 dB implies an error probability just below 10% for gain state 1. We wish to investigate how effective whacks are at revealing the appropriate gain state. The example posed is representative of the whack operation.

Below 9 dB the packet error probability at gain state 0 is about 10% or higher. The algorithm will tend to oscillate between gain states 0 and 1. As the SNRdecreases, the probability of receiving a whack while in gain state 0 increases from about 10%, reaching a maximum of about 95% around 6 dB, and then decreases to about 35% at SNR= 1dB.

As SNRdecreases towards 6 dB, successes at gain state 0 become unlikely, while whacks become very likely. Without error feedback, the algorithm would typically have to observe three unsuccessful attempts before deciding that the gain is too low. This contributes to a higher packet error probability and slower reaction time.

As SNRdecreases from 6 dB to 1 dB, it becomes more likely that the algorithm will receive no reply at gain level 0 during three transmission attempts of a packet. In the worst case, when SNRis 1 dB and the whack probability is about 35%, the probability of receiving one whack within just two transmission attempts is still about 60%. Thus, even with moderate levels of contention it is still likely that the algorithm will find out that gain state 0 is a poor choice by receiving a whack rather than by observing three consecutive attempts with no reply.

This example illustrates the usefulness of using error feedback in the channel. Whacks provide a more accurate way of discriminating between contention and channel noise over a wide range of signal-to-noise ratio values within the dynamic range of the algorithm. The gain difference between consecutive power levels (gain state 3 and lower) is 8 dB. For gain states 4 and above, the gain difference is smaller. Hence the probability of receiving a whack given that the transmission takes place at a gain state one unit too

low is even greater than for the example given above. This means that whacks are even more useful at gain states 4 and above.

Another factor that must be considered in the performance of error feedback is the packet length. When the average packet length is small, the ratio of packet length to sender identity field length is smaller than the ratio for larger packets. The probability of an error in the sender identity field approaches the probability of a packet error. The channel outcomes become either 1 or 0 only and whacks are seldom observed. For average packet lengths around 100 bits, whacks are most likely to be useful at gain states 4 and above, and only over a reduced range of signal-to-noise ratios at gain state 3 and below. Whacks are most effective for large packet traffic.

5.3 Speed of Response

Step response We consider two related measures for the speed of response of the algorithm: step response and dynamic response. The step response is a measure of the time required by the algorithm to settle to steady state operation after a sudden change of background noise in the channel. We assume that a contention-free channel exists and that the algorithm is initially in steady state. The channel noise suddenly changes to a new constant value. Without attempting to be rigorous, the step response is the number of transmissions between the change instant and the moment when the algorithm reaches steady state again.

We are interested in the worst-case step response. Since the algorithm can increase the gain state to maximum within one packet forwarding interval, the worst-case corresponds primarily to situations when the gain must be decreased. The faster response time for gain increase is considered a strength of the algorithm. Let B be the number of successful transmissions required before decreasing gain in the algorithm and $m+1$ be the number of gain states. In the worst case, a decrease from the highest to the lowest gain state, which takes approximately Bm transmissions.

Under moderate levels of contention, the algorithm is expected to adjust within three times this number of transmissions since at most three failures are ignored per packet. In the SURAP 4 architecture $B = 10, m = 6$. Thus, the algorithm has a worst case step response on the order of 60 transmissions, and on the order of 180 transmissions under moderate contention. With a throughput of 10 packets-per-second on the link, this worst-case response corresponds to less than 20 seconds, which seems appropriate for terrain-induced channel changes and mobile nodes.

Dynamic response Dynamic response is a measure of the ability of the algorithm to track time-varying changes in the channel. The rate of change could be expressed in terms of the required change, per packet transmission, in the steady-state expected value of gain state needed for acceptable performance.

If the noise level changes at a rate considerably lower than one gain state every ten packet transmissions, the algorithm is able to track the channel. If the rate of change is comparable to, or just higher, than one gain state every ten packet transmissions, the algorithm will become sluggish in tracking, especially when reducing the gain state. The result is a somewhat inefficient use of the channel. If the rate of change is about 1 gain state per packet transmission, tracking will be sluggish both in increasing and decreasing gain state. Performance will suffer from increased packet error probability and from inefficient channel use. This is probably the worst operating condition for the algorithm.

When the rate of change is considerably higher than one gain state per packet transmission each packet transmission will experience several changes in signal-to-noise ratio and may suffer from burst errors. The final effect is an equivalent channel signal-to-noise ratio, corresponding to the packet error rate induced by the burst errors. The algorithm adjusts properly to this equivalent channel noise level.

5.4 Bit Error Computations

Gain decrease When FEC is in use, the data bit error probability is lower than the channel bit error probability because of the FEC gain. When a gain state i experiences a success probability around 90% we find, through numerical evaluations, that typical channel bit error rates are on the order of 1%. Such error rates can be estimated with just a few packets, even for small-size packet traffic. After ten successful transmissions there will often be enough accumulated error data to estimate the error rate and determine if a decrease in gain state is convenient.

Looking at Figure 6.5 we see that 1% bit error probability maps to channel SNRusing the steepest of the two straight line segments in $F(\cdot)$, which has the lowest sensitivity to measurement errors in bit error probability.

Preliminary simulations using a network simulator (OpNet^{TM}) with a realistic channel model indicate that these bit error calculations are good at preventing unnecessary decreases in gain.

Gain increase Bit error calculations can lead to an increase in gain state even when an ack is received as feedback. The gain state increase is typically one unit, but could be two units. The bit error calculations are useful in two situations. First, when the signal-to-noise ratio varies slowly these calculations help trigger a gain state increase before a packet error occurs. Second, when the channel signal-to-noise ratio is appropriate for current small-size packets but would be too low for large-size packets. In this last case, if fairness is a consideration as in our algorithm, the bit error calculations are capable of detecting this situation and inducing a gain state increase.

As a concrete example consider a deteriorating channel where the signal-to-noise ratio for gain state i implies a probability of receiving an ack between 10% and 90%. In other words, gain state i is becoming an unacceptable gain state in terms of its success probability. We have found through numerical evaluations that typical *channel* error probabilities in these situations lie in the range of 1% to 10%. This channel bit error rate can be estimated within one or just a few packets, even for small-size packet traffic. An increase of one unit in the gain state is typically sufficient to provide an ack probability greater than 90%. For radio systems that provide a greater number of more closely spaced gain states, it is possible that the necessary gain increase will be two units or more. In SURAP 4, gain state increases of two units tend to occur only if the algorithm is operating at gain state 3 and the channel conditions deteriorate. A transition to gain state 5 may be in order since gain state 4 is only 0.6 dB higher than gain state 3.

Looking at Figure 6.5 we see that most of the 1% to 10% bit error range corresponds to the line segment of steepest incline in $F(\cdot)$, which has the lowest sensitivity to measurement errors in bit error probability. Simulations can help decide if a third intermediate segment is necessary to avoid estimation errors. The effect of estimation errors are considered below.

We do not use bit error calculations to decide for gain state decreases of more than one unit, since the situation typically corresponds to a very low channel bit error rate and the estimating process has

very few bits to work with or would require a large number of packets to acquire a good estimate.

5.5 Robustness

Perhaps the main factor contributing to the robustness of the Parameter Selection Algorithm is that the algorithm is designed to react to unsuccessful packet transmissions. Therefore, the algorithm can compute directly the variable we desire to control: packet error rate. Another contributing factor is the monotonicity of packet success rate and of packet overhead with gain state in the Gain Table. This means that a decision to increase gain results in a decrease in packet error probability and a decision to decrease gain results in lower packet overhead. The combination of these two factors tends to provide a stable tracking system.

The algorithm seeks to increase gain fairly rapidly if transmissions are unsuccessful, and seeks to decrease gain if the current gain seems unnecessary to maintain the desired success rate. Maintaining this success rate is given priority over reducing channel overhead due to transmissions. This means that a jammer device which does not have enough power to overcome the dynamic range of the algorithm can attempt to decrease the network's throughput but finds it hard to fool the algorithm into believing that data transmissions are successful when they are not. This latter situation would have the potential to disrupt connectivity and is considered more harmful than low throughput[2]. Moreover, since the algorithm is adaptive the jammer must remain active to force a degradation in throughput or the algorithm will revert to lower overhead transmissions.

Another problem presented by a jammer is the potential reduction in FEC gain, for example with pulse jamming. If FEC is not in use, pulse jamming and other non-stationary noise processes simply map into some packet error probability and the algorithm adjusts to it. With FEC, packet errors still lead to gain increases, but we need to look into the performance of the bit error calculations, since they utilize nominal values of FEC gain. When the true FEC gain is smaller than the nominal gain, the conclusions from bit error calculations will underestimate the gain increase required. Hence acting according to bit error calculations cannot be detrimental to the algorithm. Remember that bit error calculations never prevent packet error observations from increasing the link gain, and never trigger gain decreases by themselves. Thus, in the situation considered, we expect the algorithm to behave as if bit error calculations were not in use. More extensive research is required to properly characterize the robustness of the algorithm against special stationary and adaptive jamming strategies.

[2]Preventing masquerading and tampering with the network is left to the security protocols running in the network

6. Conclusions

6.1 Conclusions

Background The goal of the SURAP 4 architecture is to create a survivable network. In this report we have described the design, and performance of an adaptive radio-parameter selection algorithm. The objective of the algorithm is to maintain acceptable radio-link quality in the SURAP 4 packet radio architecture in the face of changing terrain conditions and jamming. The Parameter Selection Algorithm can be adapted to other packet radio environments.

The SURAP 4 radio hardware provides several choices for three important radio-channel parameters: power level, FEC rate, and channel bit rate. The main compromises the algorithm must make are among preserving good radio-link throughput, requiring low channel overhead for packet transmissions, and presenting a stable behavior to higher layer algorithms. The algorithm's primary design goal is to maintain an acceptably low packet error probability. This probability is calculated from channel noise and channel interference. The maximum acceptable value for this packet error probability has been set at 0.1 . Another design goal is to minimize the power, coding overhead, and symbol duration overhead in the channel, as long as the average packet error probability remains acceptable.

Design The goals discussed above are met by adjusting the radio parameters using feedback received through the acknowledgment radio channel. The operation of the algorithm consists of a walk among the entries in a table detailing the parameter choices. Each entry in the table specifies a power level, an FEC rate, and a channel bit rate. The steps in the table are the result of applying the transition rules in the algorithm to the feedback observed from the channel. Entries in the table are ordered in such a way that moving in one direction monotonically decreases the packet error probability, as well as monotonically increasing the channel overhead for the packet transmissions. The algorithm observes packet transmission successes and failures to decide which way to move in the table.

The Parameter Selection Algorithm uses error feedback in its decision making process. If a node receives a packet in error but is able to determine the identity of the sending node, this receiving node returns a special packet similar to the active acknowledgements used in the SURAP 4 architecture. This packet tells the sending node that its transmission did not fail because the transceiver was unavailable for reception, but probably because there was noise in the channel. The unavailability of a transceiver correlates to congestion situations and the appropriate reaction to it is typically the opposite of the action required to overcome channel noise. The error feedback mechanism was implemented to help discriminate between the two situations. The noise in the channel may consist of CDMA interference. We tolerate this ambiguity in the design of the algorithm since we expect other network algorithms to contribute in overcoming this type of interference.

The Parameter Selection Algorithm also uses bit error rate calculations in its decision making process.

The sequential decoder used for FEC encoded packets provides an estimate of the number of encoded bits received in error. The algorithm accumulates this information to compute an estimate of the channel signal-to-noise ratio. The algorithm uses a model of the post-decoding bit error probability as a function of the channel signal-to-noise ratio to calculate the required changes in the position of the algorithm in the table of radio parameters. For these calculations each parameter is assigned a nominal signal-to-noise ratio gain relative to the parameter selection offering the poorest radio link quality. The incorporation of bit error calculations in the Parameter Selection Algorithm also serves as a study on the use of bit error information for adaptive control of the radio links in a packet radio network.

Performance Simulations of the Parameter Selection Algorithm's performance on a noisy link indicate that it is capable of meeting the stated goals and that the use of error feedback and bit error calculations enhance its performance. More extensive analysis and simulation is required to certify fully the algorithm's performance, because operating conditions for the network can vary widely. One parameter can be used to adjust the average packet error probability maintained by the algorithm. This parameter is the number of consecutive successful packet transmissions that must elapse before the algorithm attempts to reduce the signal-to-noise ratio gain of its current radio parameters. The value of this parameter also affects the algorithm's step response. Hence there exists a trade-off between the average packet error probability and the algorithm's response time.

We have argued that we expect the effect of CDMA interference in the performance of the algorithm to be of secondary importance because link-layer algorithms and congestion control algorithms prevent excessive transceiver contention. To date, the results of a detailed network simulator, using OpNet™ simulation software, seem to support that this is the case. However, we do not have enough simulation data to confirm our hypothesis.

We do not rule out the possibility that network configurations will present CDMA interference problems. When this is the case, the link-layer algorithms and Congestion Control Algorithm may be a better place to look for the solution than the Parameter Selection Algorithm. More experience with the algorithm is required to determine whether this approach is effective in controlling CDMA interference.

Semi-Markov Model We have derived a Semi-Markov Model of the operation of the algorithm, which can be used to compute the steady state probability distribution of transmission gain, packet error probability, transmission power overhead, and packet duration overhead. The model is general enough to include a stationary model of transceiver blocking probability. It could also model dynamic (state-dependent) CDMA interference. Unfortunately we do not have available a characterization for this interference. The states transitions in the model could be altered to represent alternative algorithms that are based on similar operating principles.

The Semi-Markov model was introduced in Section 5.1 by modeling a specific transmission channel. Simulation results confirmed that the model gave accurate results in the example presented[1]. The specific model was generalized to allow all possible gain states and any channel signal-to-noise ratio.

The model developed omits bit error calculations. We have proposed a way to incorporate bit error calculations into the model without giving up the relative simplicity of the Semi-Markov model.

[1]Neither the model nor the simulation included the use of bit error calculations

6.2 Further Research

FEC gain Several important topics must be researched more thouroughly to create a better characterization of the Parameter Selection Algorithm. To make bit error calculations, we modeled FEC codes by a nominal, constant signal-to-noise ratio gain on the channel signal-to-noise ratio. The validity and robustness of this model must be tested against non-Additive White Gaussian noise processes on the channel. These processes can be the result of jamming actions, environmental noise, or terrain conditions, such as fading. We need improved characterizations of the operation and performance of the sequential decoders in use in the packet radio hardware because the models currently in use are inaccurate. A characterization of the estimation errors when making bit error calculations is also important. We have neglected these considerations in this report because the packet radio hardware for SURAP 4 uses interleaving and de-interleaving techniques and because the algorithm is designed to be robust against estimation and model errors expected in the bit error calculations.

Fairness vs. Throughput We expect that large packets will be more prone to noise errors than small packets since their larger length exposes them to potential sources of error for a longer amount of time. Bit error calculations allow the algorithm to enforce a fairness criterion that is aimed at guaranteeing acceptable link quality for all packet sizes. There exists a trade-off between channel overhead and the implementation of fairness. The link gain required to maintain an acceptable packet error rate for large packets represents unnecessary packet transmission overhead for small packets. This overhead takes the form of unnecessarily high power levels, coding lengths, and symbol durations. We need more experience with the algorithm to determine a good trade-off point.

CDMA interference More network simulation experience is necessary to determine conclusively the effect of CDMA interference on the Parameter Selection Algorithm. The interactions among network algorithms in a packet radio network are typically complex. Hence we believe that realistic simulations offer one of the best approaches to understanding the effect of CDMA interference, as well as the nature of interaction between the Parameter Selection Algorithm and higher-layer algorithms. We prefer to wait for more simulation experience before attempting any major changes to the transition rules in the Parameter Selection Algorithm. Altough we have been experimenting and analyzing selected modifications of the algorithm, the modifications we have tried are all relatively minor.

Semi-Markov Model It is very important that future research implements and tests the recommended changes in the Semi-Markov model of the algorithm incorporating bit error calculations. The validity of the resulting model can be checked against simulation results. The advantage of the mathematical model is that it reduces the time and effort required to characterize the steady-state performance of the algorithm.

Dynamic Response We have provided a simplified worst-case analysis of the speed of response of the algorithm against time varying channel noise. Since one important strength of the algorithm should be its ability to adapt quickly, a more thorough characterization of the algorithm response will be essential for fine tuning its design and improving its performance. The most important scenario to investigate

is a time varying noise situation in which the average time between changes is comparable to the time between packet transmissions. We expect the algorithm to exhibit its poorest adapting behavior in this situation.

Jamming Techniques It is also important to characterize the behavior of the algorithm in the face of sophisticated jamming techniques.

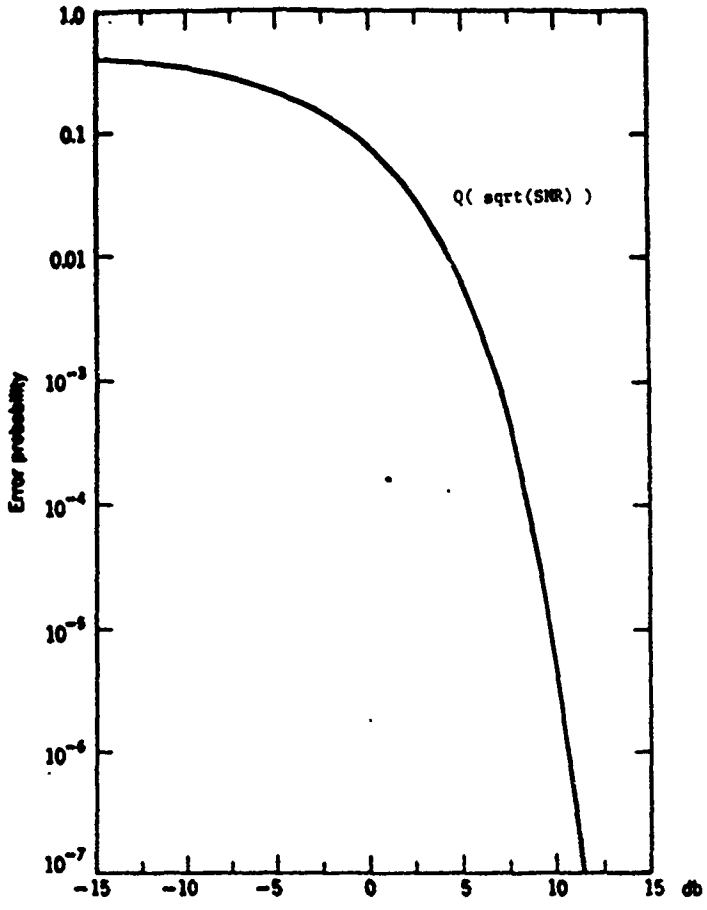

Figure 6.1: Binary Antipodal Signal Errors

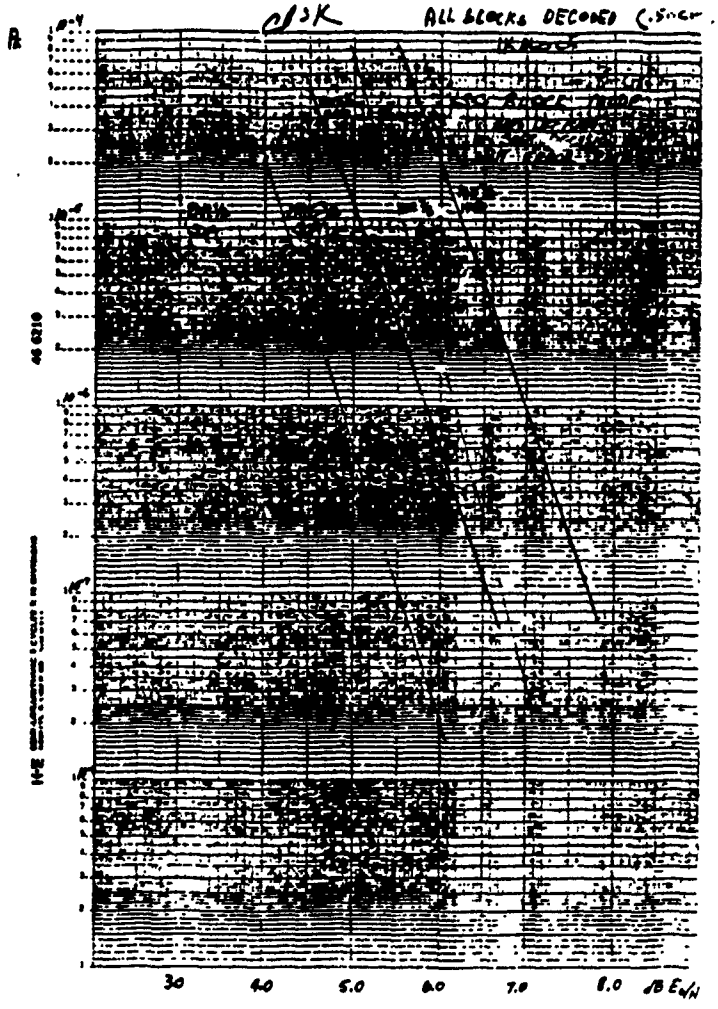

Figure 6.2: Error Performance of Sequential Decoder

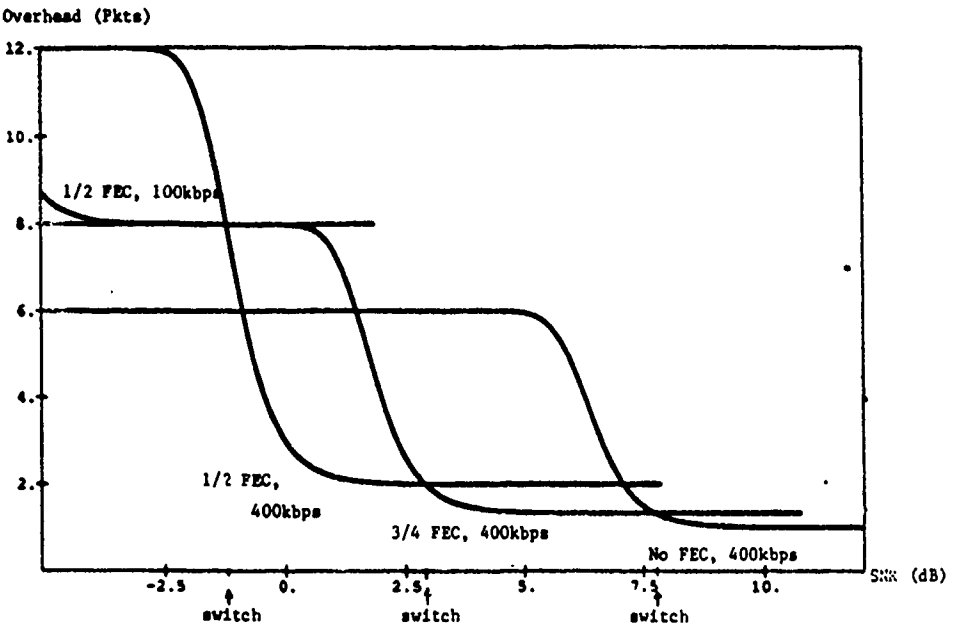

Figure 6.3: Packet Time Overhead

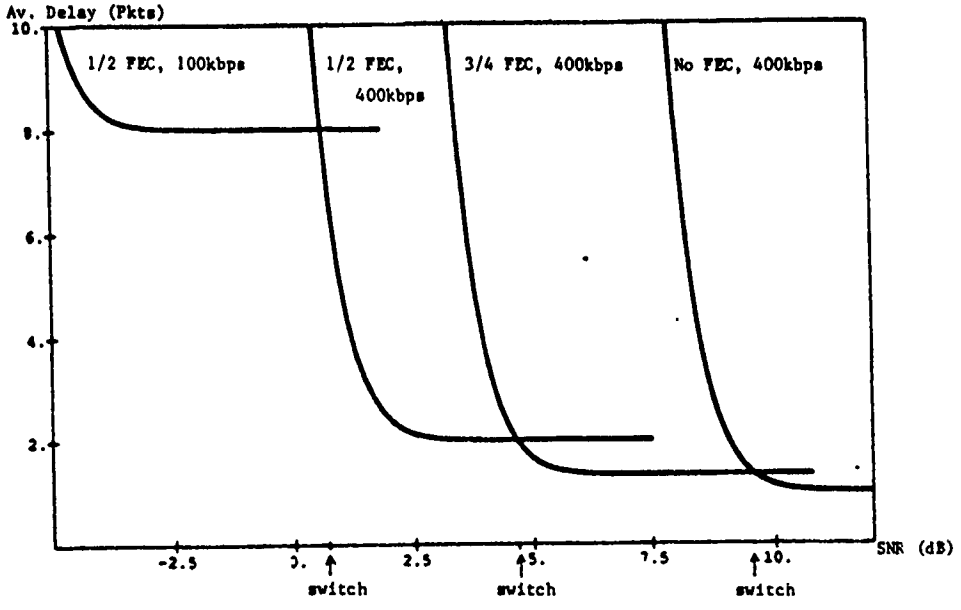

Figure 6.4: Link Forwarding Delay

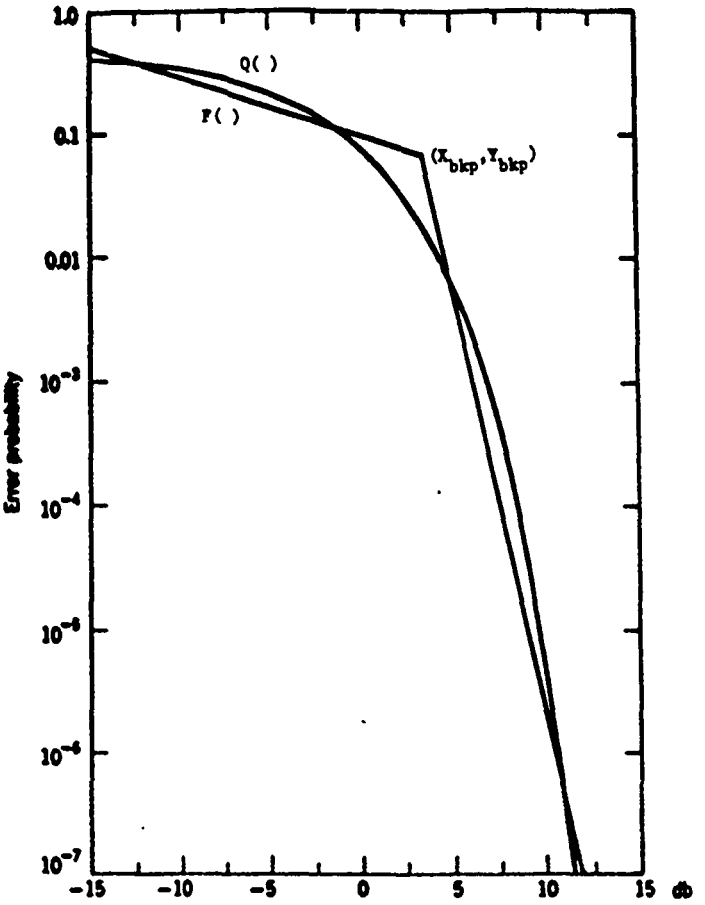

Figure 6.5: Approximation to Antipodal Signal Errors

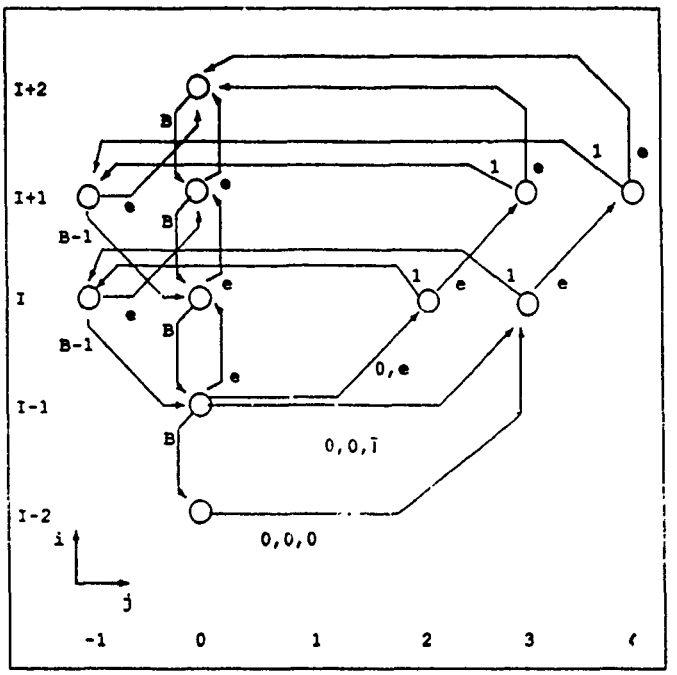

Figure 6.6: Embedded Markov Chain (specific model). Vertical dimension corresponds to gain index i. Horizontal dimension corresponds to variable j. For $j \geq 0$, j represents previous transmissions of the packet entering gain state i.

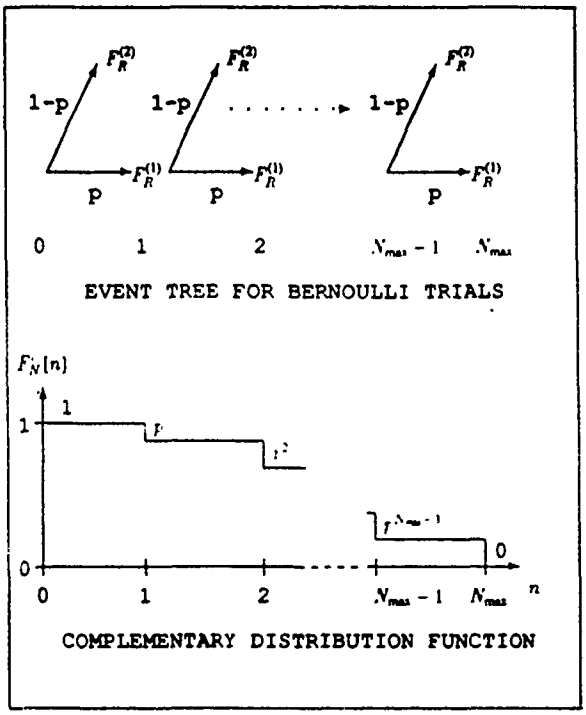

Figure 6.7: Holding Time Calculations

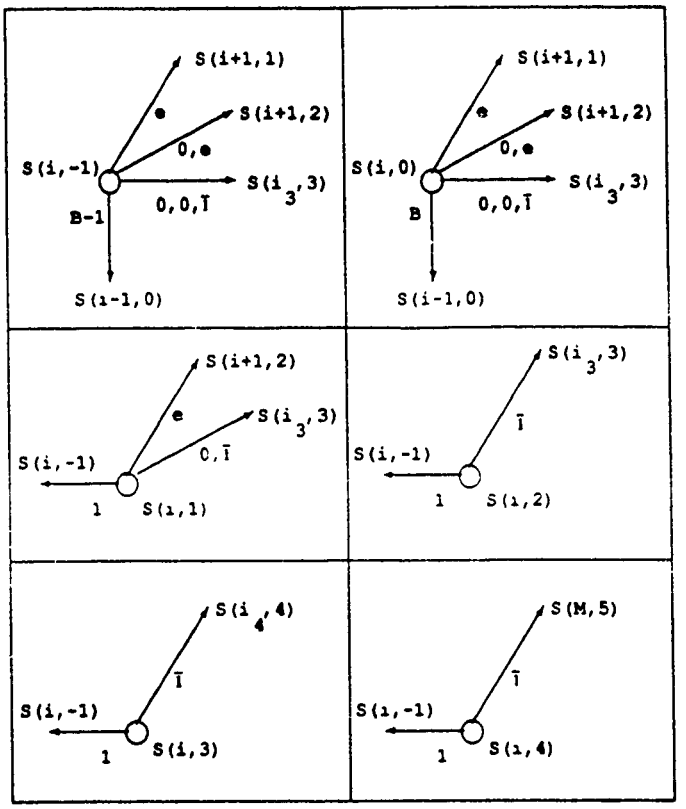

Figure 6.8: Semi-Markov Model Intermediate StatesMarkov States $S(i,j)$, $L < i < M$.

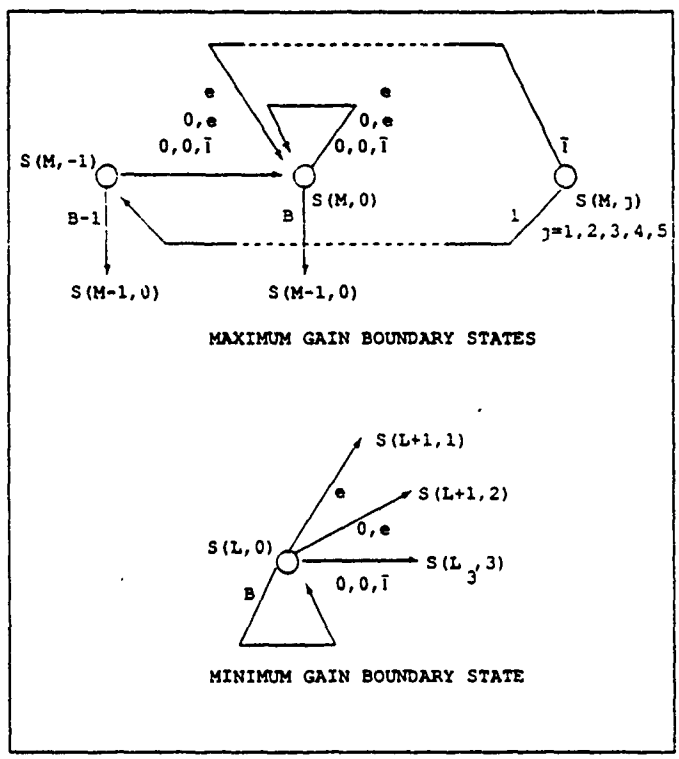

Figure 6.9: Semi-Markov Model Boundary States

Appendix A. Bit Error Function $F(\cdot)$

$F(\cdot)$ is a piecewise linear approximation to the function $Q(\sqrt{\cdot})$. They appear plotted in Figure 6.5. Let x be the signal-to-noise ratio and y be the bit error probability, expressed in dBs. (In the figure the vertical axis is not labeled in dBs.) In the expressions below, the constants recomended are: α_1 =-0.05, α_2 =-0.67, Y_{bkp} = -1.2dB, X_{bkp} = 3.5dB.

$$F(x) = \begin{cases} Y_{bkp} + \alpha_1 (X_{bkp} - x), & x \le X_{bkp}; \\ Y_{bkp} - \alpha_2 \cdot x - X_{bkp}), & x > X_{bkp}. \end{cases} \quad (A.1)$$

The inverse function $F^{-1}(y)$ is given by

$$F^{-1}(x) = \begin{cases} X_{bkp} - (y - Y_{bkp}) / \alpha_1, & y > Y_{bkp}; \\ X_{bkp} + (X_{bkp} - y) \alpha_2, & y \le Y_{bkp}. \end{cases} \quad (A.2)$$

Appendix B. Software Simulator

The simulator is written in C and currently resides and runs on the "mannix" Sun Workstation that is administered by the Survivable Adaptive Networking Group, Department 44, BBN Systems and Technologies. It simulates the operation of the algorithm on a single bidirectional link.

The simulator has a variety of capabilities. It simulates channel bit errors by applying equation (3.1) to the channel signal-to-noise ratio. This signal-to-noise ratio can include background noise, interference, and receiver noise. Each of these noise components can be time varying, even within a packet time. Data bit errors are calculated by increasing the channel signal-to-noise ratio by the FEC gain of the gain state in use. The order and combinations in the gain table of the algorithm can be arbitrary. User-supplied parameters control the following three probabilities for both the data channel and the feedback channel: packet synchronization probability, packet-header capture probability, and transceiver blocking probability. The packet sizes and the size of the sender-ID field for a packet can have arbitrary lengths as long as the latter does not exceed the former. The length of a simulation is controlled by specifying the number of data packets to be forwarded by the sending node.

The user has a choice of several software modules for the Parameter Selection Algorithm. Two of the available modules consist of the version presented in this report and of a similar version that ignores bit error calculations. We have also simulated other variants.

A set of software macros was written to collect statistics from the simulator. The software can print a complete history of the gain states chosen by the algorithm in successive transmissions. The macros can compute average packet error rate, average transmission power overhead, average packet duration overhead, and average bit error rates for the channel bits and for the data bits.

Bibliography

[1] M. Chen and R. Boorstyn. Throughput analysis of code division multiple access (cdma) multihop packet radio networks in the presence of noise. *IEEE International Conference on Communications*, 1985.

[2] O. deSouza, P. Sen, and R. Boorstyn. Congestion based routing in packet radio networks. *IEEE International Conference on Communications*, 3, Boston,1989.

[3] W. Fifer and F. Bruno. The low-cost packet radio. *Proceedings of the IEEE*, 75(1), 1987.

[4] G. Lauer. Survivable packet radio network protocols. *NATO, Shape technical center conference*, III(1):R.1–R.15, Mar 89.

[5] G. Lauer, M. Steenstrup, and J. Escobar. *A Rate-Based Congestion Control Algorithm for the SURAP 4 Packet Radio Architecture (SRNTN–72)*. Technical Report, BBN,Technical Report 7171, Cambridge,MA, December 1989.

[6] J. Lehnert. An efficient technique for evaluating direct-sequence spread-spectrum multiple-access communications. *IEEE Transactions on Communications*, COM-37(8), August 1989.

[7] J. Lehnert and M. Pursley. Error probabilities for binary direct-sequence spread-spectrum communications with random signature sequences. *IEEE Transactions on Communications*, COM-35(1), January 1987.

[8] M. Leib and J. Zavgren. *High-Throughput, Survivable Protocols for CDMA Packet Radio Networks (SRNTN–74)*. Technical Report, BBN,Technical Report 7173, Cambridge,MA, March 1990.

[9] C. McGillem and G. Cooper. *Modern Communications and Spread Spectrum*. McGraw Hill, 1st edition, 1986.

[10] N. Nazari and R. Ziemer. Computationally efficient bounds for the performance of direct-sequence spread-spectrum multiple-access communications systems in jamming environments. *IEEE Transactions on Communications*, COM-36(5), May 1988.

[11] M. Pursley, D. Sarwate, and W. Stark. Error probability for direct-sequence spread-spectrum multiple-access communications— part i: upper and lower bounds. *IEEE Transactions on Communications*, COM-30(5), May 1982.

[12] M. Pursley and D. Taipale. Error probabilities for spread-spectrum packet radio with convolutional codes and viterbi decoding. *IEEE Transactions on Communications*, COM-35(1), January 1987.

[13] P. Sen. Local markov models for multihop packet radio networks: cdma. *IEEE International Conference on Communications*, 1986.

[14] N. Shacham. Performance of arq with sequential decoding over one-hop and two-hop radio links. *IEEE Transactions on Communications*, COM-31(10), October 1983.

[15] J. Stevens. *Spatial Reuse through Dynamic Power and Routing Control in Common-Channel Random-Access Packet Radio Networks*. Technical Report, Ph.D. Thesis, University of Texas, Dallas, 1988. SRNTN-59.

[16] J. Storey and F. Tobagi. Throughput performance of an unslotted direct-sequence ssma packet radio network. *IEEE Transactions on Communications*, COM-37(8), August 1989.

[17] J. Zavgren. The moment-of-silence channel-access algorithm. *Proceedings of IEEE MILCOM89*, 2:20.3.1–20.3.7, October 1989.

[18] J. Zavgren and G. Lauer. The performance improvement from receiver-directed packet transmission in spread-spectrum packet-radio networks. *Proceedings of the 1988 Tactical Communications Conference*, 65–72, May 1988.